

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE <i>Aug 95</i>		3. REPORT TYPE AND DATES COVERED	
4. TITLE AND SUBTITLE <i>A Methodology For Assessing System Availability With Finite Queues, Component Redundancy, And Square Components</i>				5. FUNDING NUMBERS	
6. AUTHOR(S) <i>Theodore Patrick Lewis</i>					
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) AFIT Students Attending: <i>Arizona State</i>				8. PERFORMING ORGANIZATION REPORT NUMBER <i>95-033D</i>	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) DEPARTMENT OF THE AIR FORCE AFIT/CI 2950 P STREET, BLDG 125 WRIGHT-PATTERSON AFB OH 45433-7765				10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES					
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for Public Release IAW AFR 190-1 Distribution Unlimited BRIAN D. Gauthier, MSgt, USAF Chief Administration				12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words)					
<div style="text-align: center; font-size: 2em; font-weight: bold;">19960124 084</div> <div style="text-align: right; font-weight: bold;">DTIC QUALITY INSPECTED 1</div>					
14. SUBJECT TERMS				15. NUMBER OF PAGES <i>411</i>	
				16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT		

A METHODOLOGY FOR ASSESSING SYSTEM
AVAILABILITIES WITH FINITE QUEUES,
COMPONENT REDUNDANCY, AND SPARE

COMPONENTS

by

Theodore Patrick Lewis

Accession For	
NTIS	CRA&I <input checked="" type="checkbox"/>
DTIC	TAB <input type="checkbox"/>
Unannounced <input type="checkbox"/>	
Justification _____	
By _____	
Distribution / _____	
Availability Codes	
Dist	Avail and/or Special
A-1	

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

ARIZONA STATE UNIVERSITY

August 1995

A METHODOLOGY FOR ASSESSING SYSTEM
AVAILABILITIES WITH FINITE QUEUES,
COMPONENT REDUNDANCY, AND SPARE
COMPONENTS

by

Theodore Patrick Lewis

has been approved

July 1995

APPROVED:

Jeffery K. Cochran, Chairperson
J. Bert Kewels
Wayne A. Rollins
Supervisory Committee

ACCEPTED:

D. A. Rollins
Department Chairperson

Richard W. Turner
Dean, Graduate College

ABSTRACT

Logistics support has always been a key element of military combat effectiveness as well as an important element in commercial transportation systems. This research develops an original method to assess the system availability of a small number of vehicles or machines which overcomes some of the flaws of existing models. This small calling population of machines is also known as a finite-source queueing environment. The research method also incorporates backorder distributions based on sampling without replacement which is more accurate in this environment than the more common method of sampling with replacement. Although this general approach can be computationally burdensome, this work proposes two original methods to significantly reduce the number of computations necessary to find a solution. They include eliminating the tails of individual failure distributions and selectively ignoring other components in the system because of their insignificant downtime contributions. The new method also allows the user to include component redundancy and component spares. Finally, an optimization technique is developed to allow the user to optimize operating hours given a system availability goal.

ACKNOWLEDGEMENTS

I am grateful for the guidance and encouragement of my committee chairman, Dr. Jeffery K. Cochran and committee members, Dr. Dwayne A. Rollier and Dr. J. Bert Keats during this difficult learning experience.

In addition, I would like to thank Terry Moore for his Dbase programming assistance, Dana Hill for his process knowledge of Air Force systems, Capt. Pete Miyares and Smsgt. Craig Sebring (Ret) for their support as research sponsors and for getting all the data for me when I needed it. I would also like to thank Dr. Fred Lawrence for his help on the NLIP proof and Ed Yellig for his expertise on the confusing maze of mainframe computers at ASU.

I give special thanks to my loving wife, Marcia, and son, Timothy, who had to live with me through this "learning experience".

Finally, I give thanks to the Lord without whose strength none of this would have been possible.

... to grasp how wide and long and high and deep is the love of Christ, and to know this love that surpasses knowledge-- that you may be filled to the measure of all the fullness of God. Now to him who is able to do immeasurably more than all we ask or imagine, according to his power that is at work within us, to him be glory in the church and in Christ Jesus throughout all generations, for ever and ever! Amen. (Eph 3:18b-21)

TABLE OF CONTENTS

	Page
LIST OF TABLES	viii
LIST OF FIGURES	x
 CHAPTER	
1 INTRODUCTION	1
1.1 Problem Background.....	2
1.2 Other Possible Applications.....	12
1.3 Research Objective.....	15
1.3.1 Sub-Objectives	15
1.3.2 Scope and Limitations	16
1.4 Outline of Research.....	18
2 LITERATURE REVIEW	20
2.1 Inventory Measures/Models.....	20
2.2 Queueing Models.....	37
2.3 METRIC Models.....	55
2.4 Timeliness.....	78
2.4.1 Timeliness Background	78
2.4.2 Military Timeliness Issues	83
2.5 Verification And Validation.....	85
2.6 Summary.....	99
3 METHODOLOGY	100
3.1 Data Gathering.....	100

CHAPTER	Page
3.2 Model Development.....	101
3.2.1 Current Model	101
3.2.2 Alternate Modeling Approaches	102
3.3 Research Model.....	113
3.3.1 Case I - One Part With S Spares	115
3.3.2 Case II - R Transmitters With S Sp. ..	118
3.3.3 Case III - M Transmitter Units With Transmitter Or Component Redundancy ..	122
3.3.4 Case IV - M Radios With Different Components And Redundancy.....	129
3.3.5 Calculating System Availabilities	135
3.3.6 Component Failure Feedback	137
3.3.7 Summary Of Method And Preliminary Results.....	140
3.4 Computational Efficiency.....	142
3.4.1 Component Batching	147
3.4.2 Accuracy Versus Speed Considerations ..	162
3.5 Research Model As Optimizing Tool.....	164
3.6 Summary.....	166
4 RESULTS	168
4.1 Model Implementation.....	168
4.1.1 The Research Algorithm	168

CHAPTER	Page
4.1.2 Model User Issues	174
4.2 Verification And Validation.....	181
4.2.1 Verification And Validation Results ...	181
4.2.2 Research Model Analysis	194
4.2.3 Simulation Analysis Plan	203
4.2.4 Research Model And Simulation Output Compared.....	211
4.3 Timing Issues.....	257
4.4 Optimizing Tool Implementation.....	259
5 SUMMARY AND FUTURE WORK	265
5.1 Summary.....	265
5.2 Future Work.....	270
5.2.1 Research Model Enhancement	270
5.2.2 B-1 Case Study	272
REFERENCES	274
APPENDIX	
A RESEARCH MODEL CODE	284
B AIRCRAFT DATA	304
C SIMULATION MODEL CODE	313
D DATABASE MANAGEMENT SYSTEM CODE	328

LIST OF TABLES

Table	Page
1. Sample Parts Data.....	11
2. Current Model Results.....	11
3. Comparison of Aircraft Commonality.....	17
4. Summary of Safety Stock Decision Rules.....	24
5. Sample Aircraft Data.....	53
6. Comparison of Sample Model Results.....	54
7. Examples of Hypergeometric Probabilities.....	108
8. Distribution of Failures.....	117
9. Case II Sample Results.....	121
10. Set T for First Three Up and the Last One Down	127
11. Sample Probabilities of the First K Up and	129
12. Receiver Failure Probability Distribution.	134
13. Receiver Probability of First K Up Distribution ...	134
14. Growth in Computations and Run Lengths	143
15. Batch Results for 20 Components and 1 Aircraft	149
16. Batch Results for 20 Components and 3 Aircraft	150
17. Batch Results for 25 Components and 1 Aircraft	150
18. Batch Results for 25 Components and 3 Aircraft	151
19. Batch Results for 100 Components and 1 Aircraft ...	151
20. Batch Results for 200 Components and 1 Aircraft ...	152
21. Optimum Number of Batches and Batch Size	155

Table	Page
22. Aircraft Flight Profiles and Stock Level Fill	185
23. Sizes of Test Databases	195
24. No Redundancy Transient and Autocorrelation	220
25. No Redundancy Transient and Autocorrelation	224
26. No Redundancy Comparison of Results - Full Stock ..	226
27. No Redundancy Comparison of Results - Partial	227
28. No Redundancy Comparison of Results - Zero Stock ..	228
29. Redundancy Case Transient and Autocorrelation	243
30. Redundancy Case Transient and Autocorrelation	247
31. Redundancy Case Comparison of Results-Full Stock ..	249
32. Redundancy Case Comparison of Results-Partial	250
33. Redundancy Case Comparison of Results-Zero Stock ..	251
34. Timing Study Scenarios	258
35. Timing Comparison of Research versus Simulation ..	258

LIST OF FIGURES

Figure	Page
1. RSP program and concepts.....	4
2. Role of safety stock.....	22
3. EOQ stock level pattern.....	26
4. Finite replenishment-finite shortage cost diagram....	31
5. Aborescent configuration.....	36
6. Jackson Loop example.....	48
7. Indenture relationships.....	56
8. Fallacy of percent fill.....	69
9. Dyna-METRIC computation flow.....	74
10. The continuous nature of verification and valid	98
11. Typical aircraft logistics flow	114
12. Case I - One transmitter and S spares	115
13. Case II - R operating transmitters and S spares ...	118
14. Case III - M transmitter units with S spares	122
15. Transmitter redundancy	125
16. Case IV: M radio operating system	130
17. Flow of model calculations	140
18. Computation code for number/run times	144
19. Flow of model calculations (revised)	145
20. NLIP solution algorithm	158

Figure	Page
21. Plot of NLIP objective function	162
22. B52 research model input file example	170
23. B52 research model sample output	174
24. B52 Dyna-METRIC sample input file	176
25. Opening screen to Dissertation Data Aid	177
26. Utilities option opening screen	178
27. Data manipulation screen	179
28. Model selection output screen	179
29. Code to randomly select test components	196
30. DMAS B-52 scenario	198
31. Sample of DMAS computation summary	199
32. Sample of DMAS Deployment Worksheet	201
33. Time-based plot of simulation results	205
34. Plot of potential autocorrelation among the availability samples	207
35. Code to calculate the sample variance using covar .	209
36. E-3B partial stock research model output	212
37. E-3B partial stock simulation model output	213
38. Time series plot of simulation results without transient effect - F-16C run	215
39. Time series plot of simulation results with transient - E-3B run	216

Figure	Page
40. Plot of insignificant simulation autocorrelations - F-16C run	217
41. Plot of high simulation autocorrelations - F-15C ..	218
42. Elimination of transient data - E-3B run	222
43. Reduction of autocorrelation - F-15C run	223
44. Histogram of no component redundancy model diff ...	230
45. Sample mean and variance of differences	231
46. Normal probability plot - no redundancy case	232
47. F-16C full stock research model output	235
48. F-16C full stock simulation model output	236
49. Time series plot of simulation results without	238
50. Time series plot of simulation results with	239
51. Plot of insignificant simulation autocorrelations .	240
52. Plot of high simulation autocorrelations - F-15C .	241
53. Elimination of transient data - F-15C run	245
54. Reduction of autocorrelation - F-15C run	246
55. Histogram of component redundancy model diff.	253
56. Sample mean and variance of differences	254
57. Normal probability plot - redundancy case	255
58. Code to implement flying hour optimization	262
59. Relationship of flying hours and stock level	264
60. Summary of research method	267

CHAPTER 1

INTRODUCTION

You will not find it difficult to prove that battles, campaigns, and even wars have been won or lost primarily because of logistics. General Eisenhower, 1945 (Daniel, 1947)

Logistics support has always been a key element of combat effectiveness as well as an important element in commercial transportation systems. The U.S. Air Force, like any private company, has a need to manage resources efficiently and to assess the potential capabilities of their limited resources. The Air Force is interested in developing tools and methods to assess the impact of logistics on combat capability. This research will primarily be concerned with the impact of aircraft spares support on the mission success of military combat aircraft but the basic methodology is extendible to commercial transportation fleets or equipment. Several Air Force organizations have an active interest in this research since they each have responsibilities in analyzing the efficiency and effectiveness of Air Force resource allocation. They include HQ USAF/LEYS, HQ AFMC WSMIS, and HQ ACC/LGSW.

1.1 Problem Background

The following is a brief description of the logistics assessment problem. During day-to-day flight operations and training, peace-time operating stocks (POS), co-located with the unit, are used to provide aircraft spares support. When hostilities arise however, units can be deployed outside of established Air Force supply chains. In order to plan for these contingencies, additional wartime support for Air Force units is provided through the War Reserve Material (WRM) program. This program is designed to support deployed operating units and relies on prepositioning of materials based on preplanned programs and schedules (AFR 400-24, 1990). In the event of hostilities, the War Reserve Material stock is additional equipment held in reserve which supplements normal peacetime operating stocks until industrial production can sustain combat requirements. It includes spares, equipment, war consumables, medical material, weapons, and other material designated as WRM by Air Force Regulation 400-24. The Mobility Readiness Spares Package (MRSP) and the In-Place Readiness Spares Package (IRSP) are a part of the War Reserve Material program for units with aircraft,

vehicles, communication systems and other specified systems. A MRSP is defined as an air-transportable kit of critical spare parts that provides sustained operations during wartime or contingency operations when normal supply channels are interrupted or fall short of demand requirements. An IRSP performs the same support function as a MRSP but is designed for a unit who will engage the enemy from its peacetime location. These packages are meant to sustain a unit for some specified period of time (usually 30 days) without external resupply (AFR 67-1, 1993). When resupply becomes available, it comes from two sources. The first source is from repair of parts in the field and the second source arrives via resupply channels. Items that are removed from an aircraft on the flightline are sent to unit-level maintenance for repair and then returned to supply. Items broken beyond the unit's repair capability are sent to a higher repair echelon and a new part is ordered (Demmy and Hobbs, 1983). Figure 1 presents a graphic of the relationships discussed above.

RSP PROGRAM AND CONCEPTS

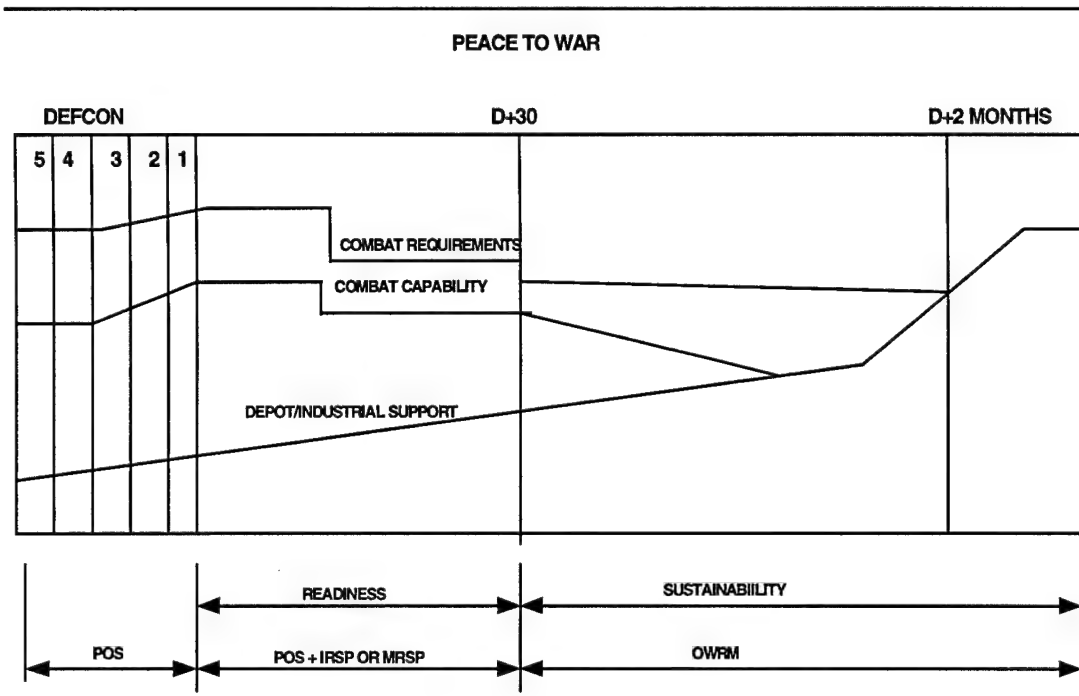


Figure 1. RSP program and concepts (Weapon System Assessment Branch, 1994)

One area of military interest is the management and assessment of the recoverable spares in the Readiness Spares Packages. Recoverable spares are those parts in the aircraft which can be repaired and reused. To model the impact of these spares on combat capability, the Air Force uses a model called Dynamic Multi-Echelon Technique for Recoverable Item Control (Dyna-METRIC) (Pyles, 1984). Dyna-METRIC is an analytical model which runs on a

mainframe computer at the Air Force Materiel Command Headquarters. Although Dyna-METRIC is a flexible tool, it has a number of limitations. This has reduced it's effectiveness in realistically predicting sortie generation capability (number of times an aircraft can fly each day) and the number of aircraft combat ready especially for units with fewer than 12 aircraft (Miyares, 1993). The model assumes, for example, that unlimited maintenance personnel and test equipment are available, that all aircraft parts are mission-essential, and that all aircraft parts fail at the same Air Force-wide flying hour-based failure rate. Also, although the model considers the impact of component redundancy, this capability has not been effectively used (King, 1993). Due to these shortcomings, the Air Force wants a more flexible method which can address the areas where Dyna-METRIC is limited.

The original work in this area assumed that all demand processes were Poisson with a mean-to-variance ratio of one. Although subsequent work has relaxed the distribution constraint to allow the use of a negative binomial or binomial distribution, the Air Force is still using the Poisson distribution for demand calculations. Since the demand distribution has little effect on the results of

this work, this research will continue to use the Poisson distribution given below:

$$p(x) = (mT)^x e^{-mT} / x! \quad x = 0, 1, 2, \dots \quad (1-1)$$

where: m - average annual demand

T - average time period

x - number of demands

The central theorem in this inventory model is Palm's theorem (Palm, 1938). This theorem is also called the infinite channel queueing assumption (Sherbrooke, 1992).

Palm's Theorem: If demand for an item is given by a Poisson process with mean m per unit time, and if the repair time for each failed unit is independently and identically distributed according to any distribution with mean repair time T , then the steady-state probability distribution for the number of units in repair has a Poisson distribution with mean mT . (Sherbrooke, 1992)

A proof of this theorem is in Chapter Two. When modeling a small number of aircraft, this theorem is violated because the demand distribution and the repair cycle are no longer independent. Consequences of this problem will be demonstrated later in an example problem.

The following mathematical development was adapted from Sherbrooke (1992). A second source for this development is Isaacson, Boren, Tsai, and Pyles (1988). The number and location of all s spare parts can be illustrated by the following equation:

$$s = OH + DI - BO \quad (1-2)$$

where:

- s - The stock level or inventory position
- OH- The number of spares on the shelf or on-hand
- DI- The number of items due-in from repair or resupply
- BO- The number of backorders for an item

This is a balanced equation because the order quantity is assumed to be one. A change in any one variable will result in a compensating change in another variable. For example, if a demand occurs, the number due-in will increase by one and, depending on the current on-hand balance, the on-hand balance will either decrease by one or the number of backorders will increase by one.

One primary use of the Dyna-METRIC model is to predict the number of aircraft available each day and the number of

sorties that can be flown during a specified time period based on an initial spares position. In order to make these predictions, the model calculates the expected number of backorders $EBO(s)$ for each item on the aircraft based on an initial stock level s for that item. This is done using:

$$\begin{aligned}
 EBO(s) &= \Pr[DI=s+1] + 2 \Pr[DI=s+2] + 3 \Pr[DI=s+3] + \dots \\
 &= \sum_{x=s+1}^{\infty} (x - s) \Pr[DI = x]
 \end{aligned}
 \tag{1-3}$$

Another common logistics term is pipeline. A pipeline represents the number of units of a part in repair at a location or in the resupply chain. The average pipeline, μ , for the single base, full repair, no depot resupply case is the average demand, m , multiplied by the repair time, T , such that $\mu = mT$. As a result of Palm's Theorem, the average pipeline value becomes the mean of the Poisson distribution for calculating the expected backorders. If we allow multiple bases, limited base repair, and depot repair and resupply, the average pipeline at base j becomes:

$$\mu_j = m_j(r_j T_j + (1 - r_j) [O_j + EBO[s_o | m_o T_o] / m_o]) \quad (1-4)$$

where m_j = average annual demand at base j

T_j = average repair time at base j

μ_j = average pipeline at base j

r_j = probability of repair at base j

O_j = average order and ship time from depot to
base j

subscript j = base counter

subscript O = depot counter

For most aircraft single unit combat assessments, this equation reduces to $\mu_j = (r_j T_j) m_j$ since depot repair and resupply are not available. As stated earlier, this value becomes the mean of the Poisson distribution used to calculate the expected backorders. Since aircraft availability (number of aircraft available versus the number of aircraft fielded times 100) is one of the important Air Force measures of merit (Pyles and Tripp, 1983), it can be calculated from the expected backorders as follows:

$$A = 100 \prod_{i=1}^I [1 - EBO_i(s_i) / (NZ_i)]^{Z_i} \quad (1-5)$$

subject to - $EBO_i(s_i) \leq NZ_i$ for every i

where Z_i - number of times the same item occurs on a
single aircraft

N - number of aircraft fielded

This formula implies that an aircraft is available only if there are no holes in any of the Z_i locations on the aircraft. This constraint simply prevents the number of backorders from exceeding the number of possible aircraft holes. The number of predicted aircraft flights/flying hours per day is simply the number of available aircraft each day times the maximum flight rate (flights/aircraft) per day which is then capped at the total number of flights required each day (i.e. Available Aircraft * max. flight rate = number of sorties flown, capped at the daily required amount).

The following numeric example displays the mathematical problems that occur when the model attempts to assess the combat capability of a small number of aircraft. Assume our unit has only one aircraft with 2 parts A and B. The following parts and scenario information is known:

Table 1: Sample Parts Data

	Part A	Part B
Demand Rate (demand/flyhr)	0.05	0.03
Percentage of Base Repair	100	100
Repair Cycle Time(days)	2	3
On-Hand Stock	0	0

Based on the above data and a requirement to fly one aircraft for eight hours every day for 10 days, the model predicts the following:

Table 2: Current Model Results

	Part A	Part B
EBOs	0.8	0.72
Pipeline	0.8	0.72
System Availability	5.6%	

These results imply that you can expect to fly 0.056 flights/day with 0.056 aircraft available on day 10 and 4.48 total flying hours for the whole 10 day period. A

simulation using the same input data predicted a daily flight capability of 0.47 flights/day and an average of 38.2 total flying hours during the 10 day period. The simulation model appears to give a much better picture of reality than the METRIC model because the METRIC model continues to break parts even when flying hours are not actually being accumulated. Next, other potential areas of application for this research will be discussed.

1.2 Other Possible Applications

The results of this research should not be restricted to military aircraft deployments. The model has potential civilian aircraft applications also. The only difficulty for civilian aircraft use is the assumption of no resupply during the assessment period. This is probably not a reasonable assumption for large U.S. commercial airlines since resupply is generally possible in a short period of time but it may be reasonable for small isolated or remote commercial or private aircraft operations especially in third world countries where logistics support is not as readily available.

For example, a small company making daily flights in a remote location could use this research method to predict

aircraft availability and the number of flights it can produce based on current on-hand stock using the resupply time of spares as the planning horizon. If the model predicts serious problems, the company could use these results to possibly justify expediting the critical parts. It would also help focus management attention on the shortfalls.

The research method could also be adapted for use with other types of vehicle transportation systems. Delivery vehicle routes are not much different than aircraft flights. The only challenge is collecting the necessary failure data and relating it to some delivery vehicle operational measure such as deliveries made, miles driven, or delivery hours. The research method is not limited to vehicles. Chapter Three illustrates how the method can be used to assess the availability of a small radio system. It could also be used for a small cellular phone network or telecommunication network.

The Army has used a METRIC-type model called SESAME (Selected Essential-Item Stockage for Availability Method) for determining spares for communications and missile equipment and directed in June 1990 that all classes of materials be provisioned using this model (Sherbrooke, 1992). There is no reason this model could not be adapted

for use in these applications as long as the resupply assumption is valid although the objective function might have to be adjusted to account for the different usage profile for a missile versus an aircraft, for example. The U.S. Navy uses a MOD-METRIC-type model called Availability Centered Inventory Model (ACIM) for ship provisioning (Sherbrooke, 1992). Again, the research model could be applied here and it seems likely that the resupply assumptions are valid especially for submarines which may not have resupply contact for up to six months at a time. This method would apply to commercial naval vessels as well as long as the resupply assumption and availability goals were reasonable.

Sherbrooke (1992) also showed how his methods could be used to calculate sparing requirements for the U.S. space station Freedom. The research methodology could be used to predict space station performance based on current on-hand stock positions. It could possibly even be extended to other inventory type systems although slight changes to the measures of merit or objective functions may be necessary to get results that make sense in the application. Below are the objectives of this research.

1.3 Research Objective

Develop a methodology to assess the impact of spares support on combat aircraft availability when deployed as a small unit with a Mobility Readiness Spares Package (MRSP).

1.3.1 Sub-Objectives

- Clearly define what a small Primary Aircraft Authorization (PAA) unit is (i.e. where do the Dyna-Metric assumptions begin to break down based on number and complexity of aircraft).
- Build a user-friendly data interface for input data manipulation.
- Document the causes of the current method's inadequacies.
- Investigate other approaches for assessing the impact of spares support on aircraft combat effectiveness.
- Develop a methodology that addresses the analytic shortcomings of Dyna-METRIC to include redundancy, item importance, and aircraft availability calculations.
- Validate the methodology using appropriate aircraft types and sizes and compare with actual experience.

1.3.2 Scope and Limitations

This research is centered around trying to find a way to more accurately predict the impact of sparing policies on the combat effectiveness of a small deployed or isolated unit. The following limits are assumed:

- 1) Only one aircraft type and unit will be analyzed at a time.
- 2) All of the aircraft are deployed or located at the same location so that transportation and ordering times of spares from supply are not significant.
- 3) Resupply from outside sources is not allowed during the analysis period or at least not from more than one source.
- 4) The parts structure has only two levels of indenture known as line replaceable units (LRU) and shop replaceable units (SRU).

These assumptions are deemed reasonable for the following reasons. Number one above is a reasonable assumption because even if more than one aircraft-type were stationed together, there is not much commonality between aircraft especially when comparisons are made between bombers and fighters (see table below).

Table 3: Comparison of Aircraft Commonality

MRSP	Line Items in MRSP	Commonality (No. of Parts)			
		F16	F15	B25	B1
F-16	173	-	8	6	1
F-15	363	8	-	3	1
B-52	347	6	3	-	10
B-1	245	1	1	10	-

Assumption number two is also reasonable since repair times are currently measured in days so unless ordering or delivery times are excessive they won't significantly impact the measurement of repair times. Also by definition, the MRSP is co-located with the unit.

Assumption three doesn't allow resupply from sources other than the unit but this requirement again comes straight from the definition of a MRSP. These spares packages were specifically designed to provide support without resupply during the support period. Based on this assumption, if resupply arrives early, the model assessment will just be a pessimistic assessment of reality. Resupply times could replace repair times as long as the resupply times are exponentially distributed with constant mean and variance.

Finally, only two levels of indenture are allowed but this is not restrictive since there are no aircraft in the Air Force inventory that allow more than two levels of indenture at the base level.

1.4 Outline of Research

This research can be broken into several parts. Chapter One introduces the topic, provides the research motivation, and sets limits on the research scope.

Chapter Two provides background material on inventory measures and models, queueing models, and METRIC-type models. It also addresses the issue of timeliness of model response and verification and validation background.

Chapter Three discusses how data was gathered for this work. Then it discusses how the research methodology was developed and presents the basic research method and model. This is the heart of the research effort. There is also a discourse on two key implementation issues. The first is how to deal with exploding computational loads and the second is how to optimally set the batching of the input data set to reduce the number of computations.

Chapter four presents the model implementation, the data analysis plans, and then the verification and

validation methods and results. It also presents a technique capable of using the research method to optimize operating hours given a user-supplied availability goal.

Chapter five presents a summary of the research results and proposes future research avenues.

CHAPTER 2

LITERATURE REVIEW

2.1 Inventory Measures/Models

There are a large number of inventory measures or goals for spares provisioning. Because of this, it is difficult to know where to start. First, some common inventory measures of merit and goals will be presented and then a brief summary of some typical inventory models will be provided.

Janson (1987) presents a number of inventory control measures he feels are important. The typical goal for each measure has also been added. They include:

- Turnover: which is the ratio of inventory investment to cost of sales. Goal: High turnover.
- Stockouts: percentage of component parts that are not available when requested or reach zero stock. Goal: Low percentage of stockouts.
- Record accuracy: cycle count to perpetual book. Goal: High record accuracy.
- Slow moving items: percentage of inventory on-hand with no orders over a given planning horizon. Goal: Small percentage of slow-moving items.

- Storage Space: percentage of open storage space to total storage space. Goal: Small percentage of empty storage space.
- Physical inventory: estimated stock compared to actual inventory. Goal: High accuracy.
- Purchased components: number of components forced to backorder. Goal: Minimize number of backorders.

All of these measures have worthy inventory goals but they are often conflicting goals. For example, eliminating slow-moving items from the inventory will improve that measure but it has the potential for increasing the number of backorders when demands come in for these slow-moving items. This action also opens up space in the warehouse increasing the percentage of empty storage space. Therefore, a company may need to prioritize their inventory goals and make trade-offs.

Greene (1974) provides another measure of inventory performance, the service index, which is the percentage of times that a customer comes in and has his requirements met. This measure is also equal to 100% minus the Stockout percentage. He also discusses the use of safety stock to buffer or protect an inventory against unexpected demands. Safety stock is stock held in addition to the requirement

necessary to support an item's average demand. The figure below shows the relationship between demand, safety stock and the reorder point.

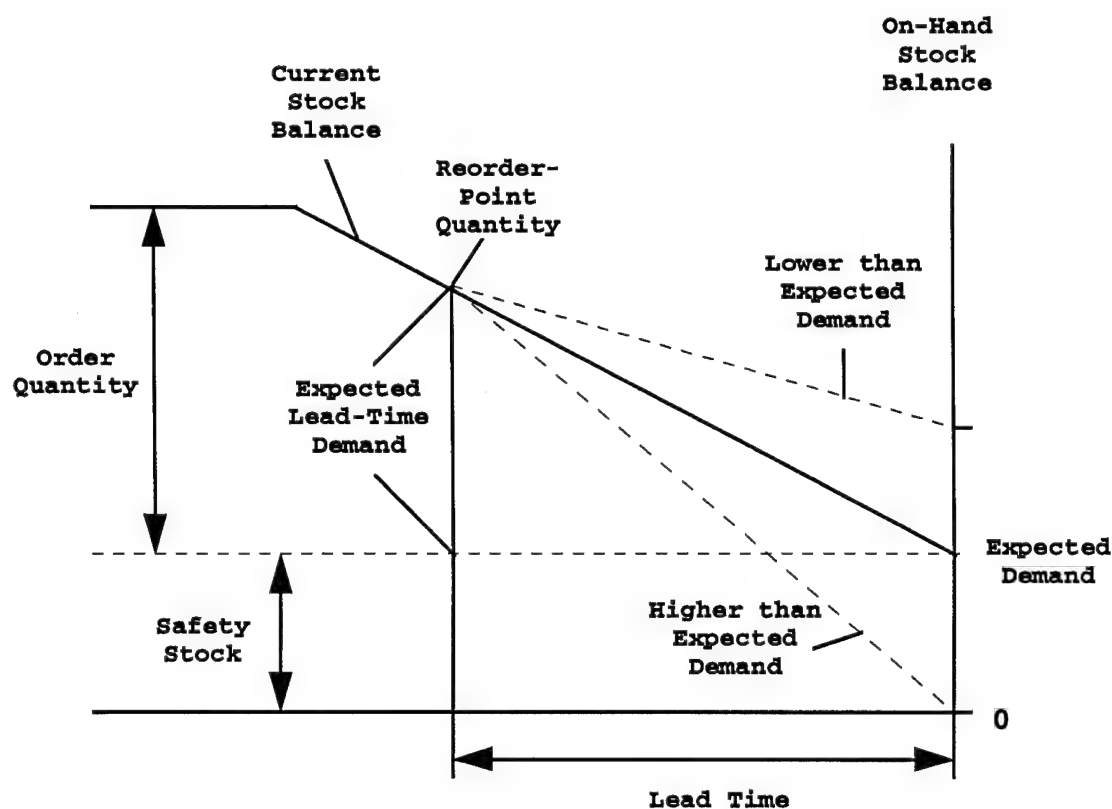


Figure 2: Role of safety stock
(adapted from Greene, 1974)

Brown (1987) presents a number of ways to compute the safety stock level depending on the goal of the inventory system. For example, to provide equal service, he sets the expected value of the safety factor to:

$$E(k) = (1 - MPV) (Q/\sigma) \quad (2-1)$$

where k is the safety factor

MPV - a management weighting factor

Q - the order quantity - equivalent to S/N or
annual sales/number of replenishment lots per
year

σ - the standard deviation of the supply lead
time

He also provides formulas to minimize backorders, minimize expediting, and minimize total cost. They are summarized in a table below:

Table 4: Summary of Safety Stock Decision Rules

(Adapted from Brown, 1987)

Objective	Formula	Management Policy Variable	Item Characteristics	Recommended
Simplicity	$k = \text{MPV}$	Safety Factor	σ	Simple systems
Minimum backorders	$F(k) = (1/\text{MPV}) N$	Expected Time between shortages	$\sigma, N, \text{budget } I$	Facing ultimate customer
Minimum expediting	$p(k) = \frac{rv\sigma}{\text{MPV} * N}$	Imputed marginal cost of shortage	$\sigma, N, I, \text{cost } v$	Intermediate echelon if expediting effective
Minimum total cost	$F(k) = \frac{rv}{N(c_2w + c_1 / b)}$	Carrying charge	$\sigma, N, I, v, \text{weight } w, \text{pieces per demand } b$	Multiple warehouse with SCOOT

v = unit cost, dollars per piece

b = pieces per demand transaction

w = unit weight, pounds per piece

 c_1 = cost per SCOOT transaction c_2 = cost per pound SCOOTed

I = Inventory budget, totaled over safety stocks for all items

 $p(k)$ = density function $F(k)$ = probability

r = carrying charge, \$/\$/year

SCOOT - Serve Customers Out Of Town - a service policy that allows order filling from alternate locations when a stockout occurs at the present location

One of the best known inventory models is the Economic Order Quantity or EOQ model. Blanchard (1986) gives a description of the basic model. This model attempts to balance the cost to hold an item in inventory with the cost to place an order for the item which results in a minimum cost strategy, assuming the model assumptions are valid. The model assumptions are given below (Lee and Nahmias, 1993):

- (1) Demand is known with certainty and fixed at D units per year.
- (2) Shortages are not permitted.
- (3) Lead time for delivery is instantaneous.
- (4) There is no time discounting of money.
- (5) Ordering costs are C_o and holding costs are C_h .

The EOQ or Q^* is:

$$Q^* = \sqrt{\frac{2C_o D}{C_h}} \quad (2-2)$$

Based on this order quantity or size, the number of orders per year (N) can be determined using:

$$N = \frac{D}{Q^*} \quad (2-3)$$

The figure below demonstrates the regular, repetitive nature of the inventory or stock level under this ordering model.

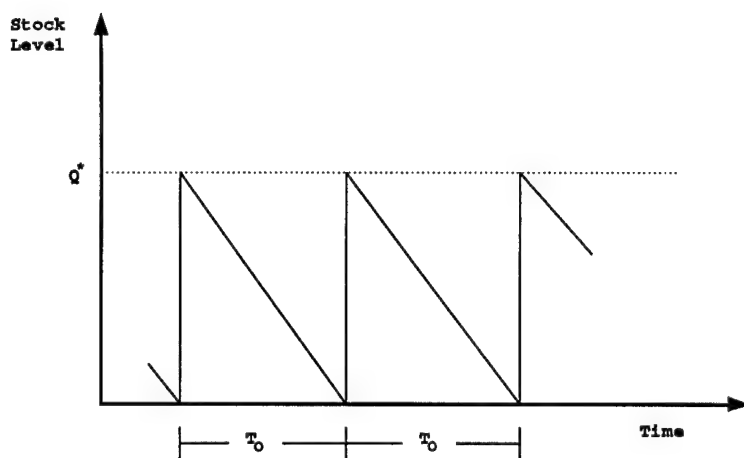


Figure 3: EOQ stock level pattern
(adapted from Waters, (1992))

The literature is filled with modifications and extensions of this basic model. One such extension of this model is found in Lee and Nahmias (1993). They give an

example for the case where the item being ordered is perishable. In this case the product is assumed to have an exponential decay or spoilage rate λ . Based on this, the on-hand inventory of the goods at time t , $X(t)$, is:

$$X(t) = X(0)\exp(-\lambda t) - (D/\lambda)[(1 - \exp(-\lambda t))] \quad (2-4)$$

This results in the following EOQ computation:

$$Q^* = \sqrt{2C_o D / ((IC * \lambda) + C_h)} \quad (2-5)$$

where IC is the unit cost of the item.

Other sources for perishable models include Nahmias (1982).

Here is a partial list of the other sources that include various modifications and extensions to the basic EOQ model: Greene (1974), Plossl (1985), Buffa and Taubert (1972), and Moore and Hendrick (1977). Almost any book on inventory theory will show this model and several variations on it.

Blanchard and Fabrycky (1990) provide a simple way to determine the appropriate quantity of spare parts for an item. They assume the item has an exponential failure rate of λ and define the Protection level, P , as the probability

of having a spare part available when required. The formula is given below:

$$P = \sum_{n=0}^s \left[\frac{(R)[-(\ln R)^n]}{n!} \right] \quad (2-6)$$

where

S = the number of spares stocked

$R = e^{-k\lambda t}$ - the composite reliability

k = the number of parts used of this type

They also showed how to determine the minimum number of spares necessary to achieve a given P value using a Spare Part Requirement nomograph.

Johnson and Montgomery (1974) present an example of a stochastic single-period model. This type of model assumes that the demand and perhaps the lead time are random variables with known probability distributions. In this situation, we are interested in determining the optimal inventory level for a single period such that the expected inventory cost is a minimum. The optimum level L^* is the solution to:

$$IC + C_h * F(L) - P * [1 - F(L)] = 0 \quad (2-7)$$

where

IC is the item cost

C_h is the holding cost

P is the sales price

$F(L)$ is the cumulative distribution for the
demand
function

They also demonstrate that this solution is a global minimum. This result gives the following inventory policy. At the end of each period, when L^* is greater than the current on-hand balance, order the difference. When L^* is less than the current on-hand balance, do not place an order.

Love (1979) contributes a model for calculating the EOQ for multiple items purchased together. This model considers the fact that some suppliers give quantity discounts on the total dollar amount purchased regardless of the mix of items. This is actually a variation of the basic EOQ model, but it tries to minimize cost based on this supplier discount. The EOQ for the group of items supplied by that business, Q_g^* , is:

$$Q_g^* = \sqrt{\frac{2 \sum C_{oj} \sum D_j}{\sum C_{hj} D_j / \sum D_j}} \quad (2-8)$$

where

C_{oj} are the ordering cost for each item j

D_j are the average demand rate for each item j

C_{hj} are the holding cost for each item j

Σ means to sum over every item in the supplier group

Banks and Fabrycky (1987) present a method that determines the optimal order quantity for a finite replenishment-finite shortage cost inventory process. This process considers the trade-off between procurement cost, holding cost, and shortage cost. It assumes that the replenishment rate exceeds the demand rate and that unsatisfied demand is not forfeited. A figure of this process is shown as follows:

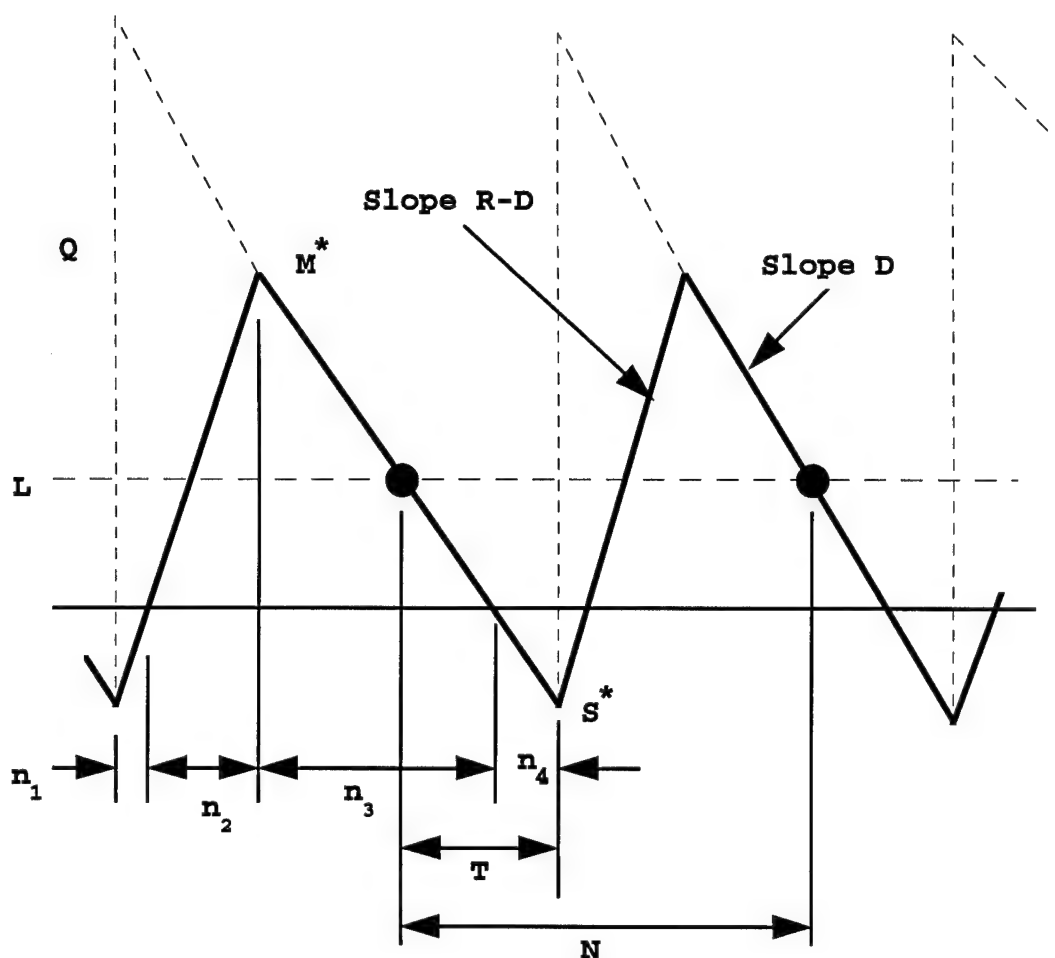


Figure 4: Finite replenishment-finite shortage cost diagram
(Taken from Banks and Fabrycky, 1987)

The maximum inventory level, M^* , for this model is:

$$M^* = Q \left(1 - \frac{D}{R} \right) + L - DT \quad (2-9)$$

where Q and D are as before

T is the number of time periods

R is the production rate in units/period

L is the procurement level.

The minimum cost procurement quantity, Q^* , is:

$$Q^* = \sqrt{\frac{2C_p D(C_h + C_s)}{C_h C_s(1 - D/R)}} \quad (2-10)$$

where the variables are defined as before.

Next they present the optimum solution when confronted with limited warehouse storage space. W is defined as the total cubic units of storage space in a warehouse. Each stock item consumes w cubic units of storage space. Defining M^* as the maximum number of items in the warehouse during any period, then M^* times w must be less than W . If M^* is less than W divided by w , then the calculations for M^* are not impacted (i.e. the computations presented earlier will suffice). If the warehouse restriction is active, then the optimum maximum inventory level is $M^* = W/w$ or:

$$M^* = Q\left(1 - \frac{D}{R}\right) + L - DT - \frac{W}{w} = 0 \quad (2-11)$$

where all variables are as before. The optimum order quantity, Q^* , then becomes:

$$Q^* = \sqrt{\frac{2C_p D}{C_s(1 - D/R)} + \frac{(C_h + C_s)(W/w)^2}{C_s(1 - D/R)^2}} \quad (2-12)$$

and the optimum procurement level, L^* , becomes:

$$L^* = DT + \frac{W}{w} - Q^* \left(1 - \frac{D}{R}\right) \quad (2-13)$$

Often times, shortage costs are unknown or unavailable so the traditional cost minimization techniques mentioned earlier are not useful. Bachmann (1986) shows how to formulate an inventory model which minimizes a stockout performance measure subject to cost and inventory constraints. He attempts to minimize the expected shortage in a year while satisfying inventory and average capital investment cost constraints. The formulation is presented below:

$$\text{Minimize } (D/Q)B(r) \quad (2-14)$$

subject to:

$$(D/Q)C_o + C_h((Q/2) + r - \lambda) \leq k$$

$$IC(Q/2 + r - \lambda) \leq m$$

where:

$B(r)$ is the function of the expected shortages

between order arrivals

r is the reorder point

λ is the lead time demand

k is the inventory cost restriction

m is the average capital investment constraint

All other variables are the same as before

In order to compute the minimum expected shortages for a year for various values of k and m , the author shows how to decouple the original formulation by using Lagrange multipliers. Then he demonstrates how to implement this method with an example from a steel mill.

Lewis (1970) presents another alternative for determining the number of replenishments based on coverage analysis techniques. He proposes that the size of replenishments be proportional to the value of the annual parts consumption. The algorithm to find the number of replenishments based on this proportionality is also presented. Based on this scheme, the order size is equal to:

$$Q_i = \frac{1}{X} \sqrt{\frac{D_i}{C_{mj}}} \quad (2-15)$$

where:

Q_i and D_i are as defined before

X is a constant of proportionality

C_{mj} is the cost of material and labor for the item

Gross, Soland, and Pinkus (1981) suggest a way to design a multi-echelon inventory system that minimizes the total expected discount inventory cost of the system. It is based on work by Clark (1958). The objective function in Clark's work was based on the recursive relation shown below for two echelons in series:

$$\begin{aligned} \hat{C}_n(x_n^1, x_n^2) = \min_{\substack{x_n^1 \leq y_n^1 \leq y_n^2 \\ y_n^2 \geq x_n^2}} \{ & c_1(y_n^1 - x_n^1) + c_2(y_n^2 - x_n^2) + L_1(y_n^1) + L_2(y_n^2) \\ & + \alpha \int_0^\infty \hat{C}_{n-1}(y_n^1 - t, y_n^2 - t) \phi(t) dt \}, \quad n = 1, 2, \dots \end{aligned} \quad (2-16)$$

where:

$c_i(Q)$ are the ordering cost functions at echelon i
with $Q \geq 0$

x_n^i and y_n^i are the inventory levels at echelon i
 before and after an order is received at the
 beginning of the n^{th} period
 $L_i(y_n^i)$ is the period cost at echelon i
 $\phi(t)$ is the probability density function of demand
 during a period
 α is the discount factor

Gross, Soland, and Pinkus (1981) then show how to adapt
 this method into a 0-1 optimization technique for designing
 large-scale inventory distribution systems. They also show
 how the model can handle systems with arborescent
 configurations (see Figure 4).

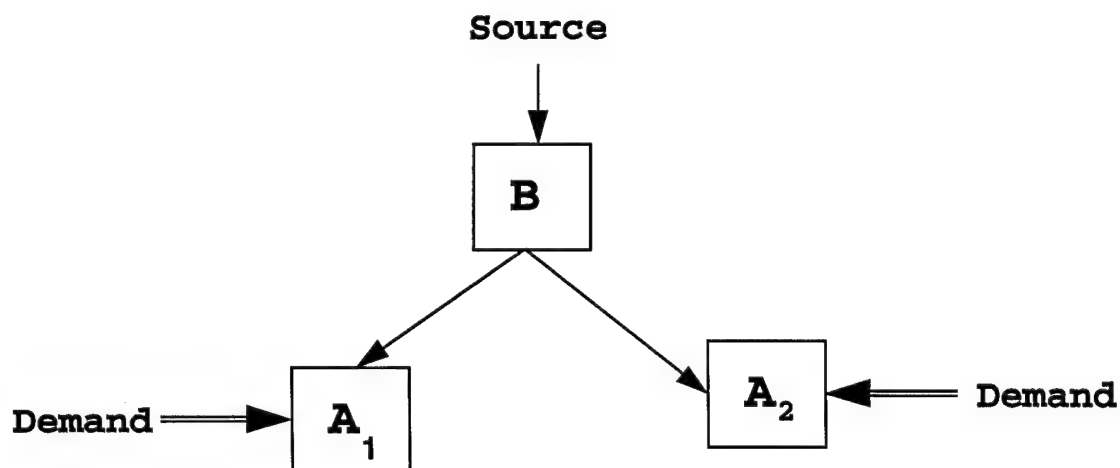


Figure 5: Arborescent configuration

(taken from Gross, Soland, and Pinkus, 1981)

They also extended the model so it can deal with system capacity constraints at the retail and wholesale levels.

Sluis (1993) presents a source for heuristic solution techniques for both deterministic and stochastic inventory problems. Having discussed a few of the many different inventory measures and models, a few of the different queueing models which might provide research leads will now be discussed.

2.2 Queueing Models

Many references exist for queueing theory and queueing models to include Perros (1994), Baccelli and Bremaud (1994), Bhat and Basawa (1992), Gnedenko and Kovalenko (1989), and Sharma (1990). Two such queueing models that offer possible solutions for this research are finite source queues and Jackson Network models. Finite source models can be found in Lee and Sengupta (1992), Balsano, Clo and Donatiello (1992), Jou, Nilsson, and Lai (1992), Chakravarthy (1992), Meyer (1993), and Takagi (1993). Sources for Jackson Networks include Gelenbe and Pujolle (1987) and Kalashnikov (1994). Both of these queueing models are presented in Gross and Harris (1985) and Medhi

(1991). Gross, Miller, and Soland (1983) also provide a more complete explanation of Jackson Networks for reliability work. The following is a brief summary of the mathematics formulas needed to perform an analysis with each of these methods. A sample analysis with each of these models for comparison with the current method and a simulation model will also be presented and discussed.

First, a discussion of finite source queues adapted from Gross and Harris (1985) will be given. Assume the calling population consists of M items. These items could be aircraft, machines, or any other such equipment. The model has a number of servers (repairmen) or repair channels C . The service times are assumed to have an exponential distribution with a mean of $1/\mu$ and operating times or arrival times of failures are also exponentially distributed with a mean time between failure of $1/\lambda$. Initially, assume there no spare parts or machines. The effective arrival rate for this system is given below:

$$\lambda_n = \begin{cases} (M-n)\lambda, & (0 \leq n < M) \\ 0 & , (n \geq M) \end{cases} \quad (2-17)$$

The effective service rate for this system is given below:

$$\mu_n = \begin{cases} n\mu, & (0 \leq n < c) \\ c\mu, & (n \geq c) \end{cases} \quad (2-18)$$

The probability, P_n , of n items or machines being in repair is:

$$P_n = \begin{cases} \frac{M! / (M-n)!}{n!} \left(\frac{\lambda}{\mu}\right)^n P_0, & (0 \leq n < c) \\ \frac{M! / (M-n)!}{c^{n-c} c!} \left(\frac{\lambda}{\mu}\right)^n P_0, & (c \leq n \leq M) \end{cases} \quad (2-19)$$

where P_0 is the probability that 0 items will be in-repair. The expression for P_0 is given below based on the fact that the probabilities must sum to zero:

$$P_0 = \left[\sum_{n=0}^{c-1} \binom{M}{n} \left(\frac{\lambda}{\mu}\right)^n + \sum_{n=c}^M \binom{M}{n} \frac{n!}{c^{n-c} c!} \left(\frac{\lambda}{\mu}\right)^n \right]^{-1} \quad (2-20)$$

There is an important invariant property for finite source queueing systems with exponential service: equations 2-16 and 2-17 are valid regardless of the form of the distribution of time to breakdown as long as the arrival rates are independent with mean $1/\lambda$ (i.e. any G/M/C model with finite source). A proof of this property is available in Bunday and Scranton (1980). The average number of items down or broken, L , then becomes:

$$L = P_0 \left[\sum_{n=0}^{c-1} n \binom{M}{n} \left(\frac{\lambda}{\mu}\right)^n + \frac{1}{c!} \sum_{n=c}^M n \binom{M}{n} \frac{n!}{c^{n-c}} \left(\frac{\lambda}{\mu}\right)^n \right] \quad (2-21)$$

and the number of items awaiting service, L_q is:

$$L_q = L - c + P_0 \left[\sum_{n=0}^{c-1} (c - n) \binom{M}{n} \left(\frac{\lambda}{\mu} \right)^n \right] \quad (2-22)$$

The average downtime in the system is also important. The following results are based on the application of Little's formula, $L = \lambda' W$, where the mean arrival rate to the system, λ' , becomes $\lambda' = \lambda(M - L)$. The average waiting time to be repaired, W_q , then becomes:

$$W_q = L_q / \lambda(M - L) \quad (2-23)$$

and the total time in system or total downtime, W , is:

$$W = L / \lambda(M - L)$$

$$\text{or} \quad W = W_q + 1/\mu \quad (2-24)$$

If Y spares are then introduced into the system, the formulas don't change significantly. The arrival rate, λ_n , becomes:

$$\lambda_n = \begin{cases} M\lambda, & (0 \leq n < Y) \\ (M - n + Y)\lambda, & (Y \leq n < Y + M) \\ 0, & (n \geq Y + M) \end{cases} \quad (2-25)$$

The effective service rate is the same as above since the service rate hasn't changed with the addition of spare parts. It does, however, impact the probability distribution of items in repair. The number of service channels and spares also impacts the distribution. If the number of channels c is less than or equal to the number of spares Y , the result is given below:

$$P_n = \begin{cases} \frac{M^n \lambda^n}{n! \mu^n} P_0, & (0 \leq n < c) \\ \frac{M^n}{c^{n-c} c!} \left(\frac{\lambda}{\mu}\right)^n P_0, & (c \leq n < Y) \\ \frac{M^Y M!}{(M - n + Y)! c^{n-c} c!} \left(\frac{\lambda}{\mu}\right)^n P_0, & (Y \leq n \leq Y + M) \end{cases} \quad (2-26)$$

If the number of channels or repairmen, c , exceeds the number of spares Y , the distribution is:

$$P_n = \begin{cases} \frac{M^n \lambda^n}{n! \mu^n} P_0, & (0 \leq n \leq Y) \\ \frac{M^Y M!}{(M - n + Y)! n!} \left(\frac{\lambda}{\mu}\right)^n P_0, & (Y + 1 \leq n < C) \\ \frac{M^Y M!}{(M - n + Y)! c^{n-c} c!} \left(\frac{\lambda}{\mu}\right)^n P_0, & (C \leq n \leq Y + M) \end{cases} \quad (2-27)$$

An expression for P_0 can then be found by using the fact

that $\sum_{n=0}^{Y+M} P_n = 1$ must be true. The number broken in this

system, L , becomes:

$$L = \sum_{n=0}^{M+Y} nP_n \quad (2-28)$$

and the number waiting to be repaired, L_q , is:

$$L_q = \sum_{n=c}^{M+Y} (n - c)P_n \quad (2-29)$$

The formulas for W and W_q can be developed in a similar matter.

A sample analysis with this model is presented below:

Sample Problem

This problem includes two parts that make up one machine with one repairman and the following data:

$c = 1$ repair channel	8 hr Maintenance shift
$M = 1$ machine	8 operating Hours/Day

Part 1

Demand Rate = 0.05 demands/operating hour = 0.4 demands/day

Mean time to repair (MTTR) = 2 days/repair or

Repair rate = 0.5 rep/day

and $\lambda / \mu = 0.8$

$$P_0 = \left[\binom{1}{0} (0.8)^0 + \sum_{n=1}^1 \binom{1}{1} \frac{n!}{1} (0.8)^n \right]^{-1}$$

$$P_0 = [1 + 0.8]^{-1} = 0.5555$$

$$L = P_0 \left[0 + \frac{1}{1} \sum_{n=1}^1 n \binom{1}{1} \frac{1}{1} (0.8)^1 \right] = 0.4444$$

Available Machines = $M - L = 1 - 0.4444 = 0.5556$ machines

L_q has no meaning since there is only one machine

$$W = \frac{0.4444}{0.4(1 - 0.4444)} = 1.9996 \text{ days} \approx 2 \text{ days}$$

Part 2

Demand Rate = 0.03 demands/operating hour = 0.24 demands/day

MTTR = 3 days/repair or Repair rate = 0.3333 rep/day

and $\lambda / \mu = 0.72$

$$P_0 = \left[\binom{1}{0} (0.72)^0 + \sum_{n=1}^1 \binom{1}{1} \frac{n!}{1} (0.72)^n \right]^{-1}$$

$$P_0 = [1 + 0.72]^{-1} = 0.5814$$

$$L = P_0 \left[0 + \frac{1}{1} \sum_{n=1}^1 n \binom{1}{1} \frac{1}{1} (0.72)^1 \right] = 0.4186$$

Available Machines = $M - L = 1 - 0.4186 = 0.5814$ machines

L_q has no meaning again

$$W = \frac{0.4186}{0.24(1 - 0.4186)} = 3.00 \text{ days}$$

One advantage to this method is that it considers the impact of a limited calling population which is a problem with the current method. However, this approach also has its limitations. For example, there is not any clear way to handle more than one part-type at a time other than to simply multiply the part-type availabilities together. This method of just combining the failure rates and using a weighted average of the base repair cycle time is demonstrated below:.

Both Parts at the same time

Demand Rate = 0.08 demands/operating hour = 0.64 demands/day

MTTR = 2.375 days/repair (weighted average of two parts) or

Repair rate = 0.4211 rep/day

with $\lambda / \mu = 1.52^*$

$$P_0 = \left[\binom{1}{0} \left(\frac{0.64}{0.4211} \right)^0 + \sum_{n=1}^1 \binom{1}{1} \frac{n!}{1} \left(\frac{0.64}{0.4211} \right)^n \right]^{-1}$$

$$P_0 = [1 + 1.52]^{-1} = 0.3968$$

$$L = P_0 \left[0 + \frac{1}{1} \sum_{n=1}^1 n \binom{1}{1} \frac{1}{1} \left(\frac{0.64}{0.4211} \right)^1 \right] = 0.6032$$

Available Machines = $M - L = 1 - 0.6032 = 0.3968$ machines

*Note: This violates the steady state requirement of $\lambda / \mu > 1.0$.

Two issues arise with this approach. If only one repair channel is used, then ρ is greater than 1 and there is no longer a steady state solution. If the modeler allows the repair channels to be greater than one, the mathematics break down because the number of repair channels should not exceed the calling population size.

Another problem with this method is that all parts are assumed to be repairable. In the real world, of course, this is not the case and the impact of one part's failure on the failure rate of other parts is not assessed. For example, if part A forces the aircraft to sit on the ground, then the demand rate for part B is zero even though part B is not broken. This method also doesn't consider how to consolidate the failures of different parts to the fewest number of grounded aircraft. A final issue relates to how this method would handle a changing arrival rate which is independent of the number of aircraft deployed (i.e. a changing flying schedule). Due to the relative brevity of deployments, the existence of unrepairable parts, and the high sortie rates and hours which result in

high arrival rates, it is unlikely that a steady state solution will exist for most of the scenarios considered.

Next, the Jackson Network Approach will be considered. It suffers from some of the same problems that the Finite Source approach does but it may be more adaptable than the previous method to the current research problem. First, formula (4-13) for computing $a_i(n_i)$ in Gross and Harris's book (1985) is incorrect. The correct formula is in the appendix of Gross, Miller, and Soland (1983) and is presented below. This presentation will center on closed Jackson Networks based on work from the two previously mentioned works. This network approach assumes that the system is closed to outside influence and, therefore, uses a finite-source queue of items which flow inside the network. Using node flow-balance equations for this situation (i.e. flow into node i must equal flow out of node i),

$$\mu_i \rho_i = \sum_{j=1}^K \mu_j r_{ji} \rho_j \quad (2-30)$$

where

$$\rho_i = \frac{\lambda_i}{\mu_i}$$

it can be shown that the probability of being in each state, if only one server or channel is allowed at each node or $c_i = 1$ for all i , is:

$$P_{n_1, n_2, \dots, n_k} = \frac{1}{G(N)} \rho_1^{n_1} \rho_2^{n_2} \dots \rho_k^{n_k} \quad (2-31)$$

where

$$G(N) = \sum_{n_1 + n_2 + \dots + n_k = N} \rho_1^{n_1} \rho_2^{n_2} \dots \rho_k^{n_k} \quad (2-32)$$

If the method is extended to allow any c_i servers or channels at each node i , then the solution becomes:

$$P_{n_1, n_2, \dots, n_k} = \frac{1}{G(N)} \prod_{i=1}^k \frac{\rho_i^{n_i}}{a_i(n_i)} \quad (2-33)$$

where

$$G(N) = \sum_{n_1 + n_2 + \dots + n_k = N} \prod_{i=1}^k \frac{\rho_i^{n_i}}{a_i(n_i)} \quad (2-34)$$

and

$$a_i(n_i) = \begin{cases} n_i ! & , (n_i < c_i) \\ c_i^{n_i - c_i} c_i ! & , (n_i \geq c_i) \end{cases} \quad (2-35)$$

To place this method in a context, assume that the network is a three-echelon repair system like the one depicted in Figure 5.

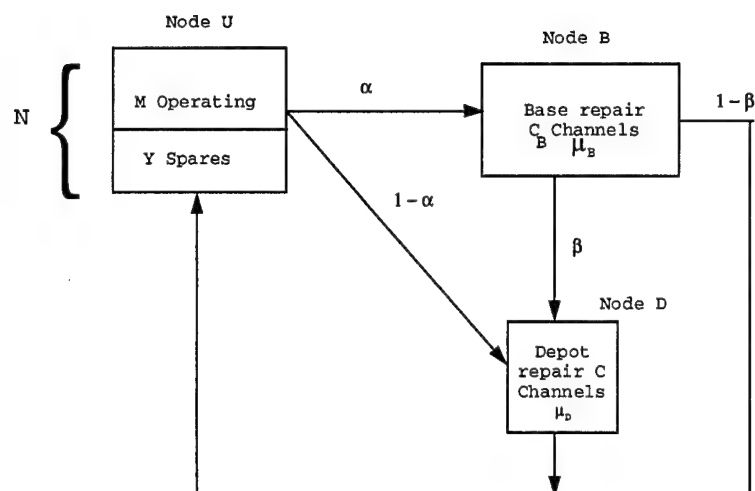


Figure 6: Jackson Loop example
(Gross, Miller, and Soland, 1983)

Gross, Miller, and Soland (1983) use this network approach to determine the optimum number of spares and repair channels in order to minimize the system cost. The minimum cost formulation for the figure above is presented below:

$$\begin{aligned}
 &\underset{Y, C_B, C_D}{\text{Minimize}} && Z = K_Y Y + K_B C_B + K_D C_D \\
 &\text{Subject to:} && \sum_{n=M}^{M+Y} P_n \geq A
 \end{aligned} \tag{2-36}$$

where:

P_n - Steady state probability that n units are operational

M - Number of components desired to be operational

A - Minimum percentage of time all M components are operational

Y - Number of spare components to stock

c_b - Base repair capacity in number of channels

C_D - Depot repair capacity in number of channels

K_i - Cost per unit ($i = y, D, B$) including annual operating costs and capital investment amortization of a spare or a repair channel.

This approach was designed to compute the minimum cost spares levels and repair channel costs for a single item based on predetermined availability goals. To solve the rest of this problem, Gross, Miller, and Soland (1983) would use Buzen's (1973) algorithm to find the normalizing constants and Lawler and Bell's (1966) optimizing algorithm to locate the optimum solution. The object of this research is to find the maximum availability, given a set of spares levels. This presents an obvious disconnect. This method also requires steady state conditions which include the assumption that flow into a node will equal flow out of a node. This will not be true if base repair

capability is not 100%. Again, the method seems limited to computing spares levels one at a time. With the research goals in mind, a reformulation of the problem was attempted. This reformulation is shown below:

$$\text{Maximize } Z = \prod_{i=1}^P \left(\sum_{n=M}^{M+y_i} P_{in} \right) = \text{Availability} \quad (2-37)$$

Subject to: $y_i \leq \text{Current stock}$

$C_{bi} \leq \text{Base repair channels}$

This determines an optimum set of P_{in} values rather than computing optimal spares levels.

The following is a sample computation using the same data provided earlier in the finite source queue example.

$$\text{Maximize } Z = \prod_{i=1}^P \left(\sum_{n=M}^{M+y_i} P_{in} \right) = \text{Availability}$$

$M = 1$ - Number of operating aircraft

$P = 2$ - Number of parts

Subject to: $y_1 \leq 0$

$y_2 \leq 0$

$C_{b1} \leq 1$

$C_{b2} \leq 1$

node 1 $\rightarrow n_1$ - is an operational aircraft

node 2 $\rightarrow n_2$ - is a broken aircraft

$$\text{The routing matrix - } R = \begin{matrix} & \begin{matrix} 1 & 2 \end{matrix} \\ \begin{matrix} 1 \\ 2 \end{matrix} & \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \end{matrix}$$

$$a_i(n_i) = \{ n_i! , (i = 1, 2) \}$$

$$\lambda_1 \rho_1 = \mu_2 r_{21} \rho_2$$

$$\mu_2 \rho_2 = \lambda_1 r_{12} \rho_1$$

$$\text{Set } \rho_2 = 1, \text{ thus } \rho_1 = \frac{\mu_2 r_{21} \rho_2}{\lambda_1}$$

$$P_{n_1, n_2} = \frac{1}{G(1)} \left(\frac{\mu_2 r_{21} \rho_2}{\lambda_1} \right)^{n_1} \left(\frac{1}{a_i(n_i)} \right) (1)^{n_2}$$

where $(n_1, n_2) = (1, 0)$ and $(0, 1)$.

For Part One

$$P_{n_1, n_2} = \frac{1}{G(1)} \left(\frac{(0.5) (1) (1)}{0.4} \right)^{n_1} \left(\frac{1}{1} \right) (1)^{n_2}$$

$$G(1) = \left(\frac{0.5}{0.4} \right) + 1 = 2.25$$

$$P_{1,0} = (1/2.25) (0.5/0.4) = 0.5555$$

$$P_{0,1} = (1/2.25) (1) = 0.4444$$

For Part Two

$$P_{n_1, n_2} = \frac{1}{G(1)} \left(\frac{(0.3333) (1) (1)}{0.24} \right)^{n_1} \left(\frac{1}{1} \right) (1)^{n_2}$$

$$G(1) = \left(\frac{0.3333}{0.24} \right) + 1 = 2.3888$$

$$P_{1,0} = (1/2.3888) (0.3333/0.24) = 0.5814$$

$$P_{0,1} = (1/2.3888) (1) = 0.4186$$

The Availability, Z, becomes:

$$Z = P * P = 0.5555 * 0.5814$$

$$= 0.3230 \text{ or } 32.30\%$$

One final observation about the two methods discussed above: neither directly addresses the issue of redundancy. A comparison of the two methods discussed above, the current method, and a simulation of a deployment scenario is presented below.

The following numeric example compares the mathematical solutions of four different approaches to assess the combat capability of a small number of aircraft. The solutions for the finite source and Jackson network

were shown earlier. The Dyna-METRIC results come from a run of the Dyna-Metric Microcomputer Assessment System (DMAS) (DRC, 1993a) with the input data presented below. The simulation model is a SLAM II (Pritsker, 1986) model written by the author again to duplicate the environment presented below. It is based on the simulation model in Lewis (1987). The simulation model for this simple case study should provide the most realistic results and, therefore, becomes the basis of comparison between the other models. Assume our unit has only one aircraft with 2 parts A and B. The following information is known:

Table 5: Sample Aircraft Data

	Part A	Part B
Demand Rate (demand/flyhr)	0.05	0.03
Percentage of Base Repair	100	100
Repair Cycle Time(days)	2	3
On-Hand Stock	0	0

The aircraft is required to fly one eight hour mission every day for 10 days. The model comparisons are in the table below:

Table 6: Comparison of Sample Model Results

Method	Sorties over 10-day period	Availability
Simulation	4.85	28.7%
Dyna-METRIC	0.56	5.6%
Modified Finite Source	N/A	39.68%*
Reformulated Jackson Network	3.22	32.30%

* Violates Steady State Conditions

The finite source method computed an availability for Part A of 55.56% and Part B of 57.89%. The method used to combine them gives an availability of 40.6% but it violates the steady state condition. The reformatted Jackson Network resulted in an aircraft availability of 32.16% with a 10 day sortie capability of 3.22 sorties. This is much closer to the simulation results than the current method. It shows promise. It still assumes independence between different component's failures and repair channels but it handles multiple parts and a finite calling population and may be able to handle the condition of less than 100% base repair. Issues of steady state conditions and redundancy will still have to be addressed. Now let's look at the development of the current Dyna-METRIC model and its use.

2.3 METRIC Models

This section looks at the history of multi-echelon, multi-indenture inventory models with an emphasis on the development of the Dynamic Multi-Echelon Technique for Recoverable Item Control (Dyna-METRIC) model that is currently being used by the Air Force to assess weapon system wartime capabilities but first, a few helpful definitions. The term multi-indenture signifies that a component has several levels or layers of sub-components. The following figure shows how levels of indenture could be described or represented.

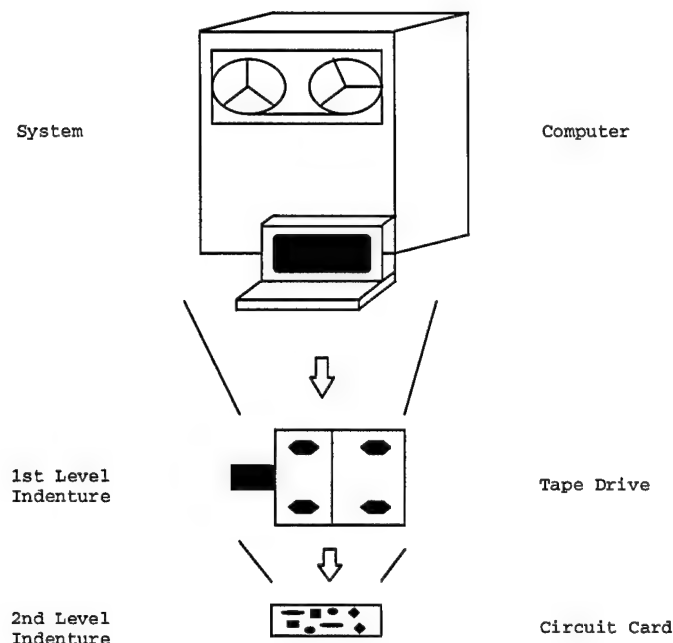


Figure 7: Indenture relationships

The Air Force refers to the item at the first level of indenture or the item that is pulled directly off the aircraft by a technician as a Line Replaceable Unit or LRU. This might include things such as radios, flight computers, or electronic countermeasures equipment. When the equipment is taken to the technician's shop for repair, the black box is opened and parts such as circuit cards are removed and replaced to repair the black box. These items that are removed and repaired or replaced in the repair shops are known as Shop Replaceable Units or SRUs. Multi-echelon is usually thought of in terms of wholesale or retail levels or levels of repair. For example, the simple

removal and replacement of auto parts at a typical gas station repair shop is one echelon of auto repair. The next higher level of auto repair would be the repair or refurbishment of these replaced parts and the final echelon might be the repair or replacement of the replacement components inside the originally defective item. Now, let us go on to a discussion of the work in this area.

The basic theorem for most of the work in this area is Palm's Theorem on queueing. This theorem is stated below:

Palm's Theorem: If demand for an item is given by a Poisson process with mean m per unit time, and if the repair time for each failed unit is independently and identically distributed according to any distribution with mean repair time T , then the steady-state probability distribution for the number of units in repair has a Poisson distribution with mean mT . (Sherbrooke, 1992)

This theorem is important because it states that the shape of the repair time distribution with mean T has no impact on the steady-state probability distribution of the number of units in the repair channel. Thus, the steady-state distribution of the number of items in the repair channel is Poisson with mean mT (Sherbrooke, 1992). The following proof of Palm's Theorem is adapted from Hadley and Whitin (1963) and presented below:

PROOF:

Assume $r(p)$ is the probability that the repair time is p when the mean repair time is T . The probability of a unit completing repair at time t , having started at t_i where $t > t_i$ is shown below:

$$R(t - t_i) = \int_0^{t-t_i} r(p) dp \quad (2-38)$$

If at least one demand has occurred in the interval 0 to t , then the probability that a unit is repaired by t is:

$$[R(t - t_i) / t] dt_i \quad (2-39)$$

The probability that a specific demand between 0 and t is repaired by t is given by:

$$\left[1 / t\right] \int_0^t R(t - t_i) dt_i = \left[1 / t\right] \int_0^t R(p) dp \quad (2-40)$$

Let s be the number of items available initially and n be the net inventory where the net inventory is the current stock balance minus any backorders. We want to show that as t goes to infinity, the number of items in repair has a Poisson distribution with mean mT . First, we need the probability that the net inventory at time t is n when there have been z demands since time 0. The values of n range from s to $s - z$ where $n = s$ means all z items have been repaired and $n = s - z$ means that none of the z items have been repaired. This implies that $s - n$ items still need to be repaired at time t .

If there have been z demands, there must be $z + n - s$ finished repairs. The following binomial distribution represents the probability that $s - n$ repairs have yet to be completed from the z demands:

$$\text{bin}(n) = \binom{z}{s-n} \left[(1/t) \int_0^t R(p) dp \right]^{z+n-s} \left[(1/t) \int_0^t \{1 - R(p)\} dp \right]^{s-n} \quad (2-41)$$

By weighting (2-16) by the probability of having z items demanded and summing over all $z \geq s - n$ we get the unconditional probability that the net inventory is n at time t or:

$$\begin{aligned}
 \Pr[\text{net inventory is } n \text{ at time } t] &= \sum_{z=s-n} p(z|mT) \text{bin}(n) \\
 &= [1/(s-n)!] \left[m \int_0^t \{1-R(p)\} dp \right]^{s-n} e^{-m \int_0^t \{1-R(p)\} dp} \quad (2-42)
 \end{aligned}$$

Since we are interested in the limit of $\Pr[\text{net inventory is } n \text{ at time } t]$ as $t \rightarrow \infty$, notice that:

$$\begin{aligned}
 \lim_{t \rightarrow \infty} \int_0^t \{1-R(p)\} dp &= \int_0^\infty \{1-R(p)\} dp \\
 &= \int_0^\infty -p \, d[1-R(p)] \\
 &= \int_0^\infty p \, (dR / dp) dp \\
 &= \int_0^\infty p \, r(p) dp = T \quad (2-43)
 \end{aligned}$$

Substituting this into equation (2-17) gives:

$$\lim_{t \rightarrow \infty} \Pr[\text{net inventory is } n \text{ at time } t] = \text{poisson}\{s-n|mT\} \quad (2-42)$$

Because $s - n$ is the number of items being repaired with a range of 0 to ∞ , Palm's Theorem is proven.

The research in this area, according to Demmy and Hobbs (1983), began with the work of Feeney and Sherbrooke (1966) based on optimization techniques for stationary, multi-echelon, multi-indenture inventory and repair systems which use $(s-1,s)$ inventory policies. Feeney and Sherbrooke describe the $(s-1,s)$ inventory system in their work. Their results will allow item demand patterns with any compound Poisson distribution. Inventory performance in this situation is dependent solely on the spare stock level, s , which provides protection against stockouts. If delay times such as resupply or repair were zero, then spare stock levels would be unnecessary. This scenario, however, is seldom true so optimal sparing levels are usually greater than zero. They define the $(s-1,s)$ policy as a continuous review policy where a demand for D units results in an immediate reorder of D units. This keeps the total on-hand stock plus stock on-order minus backorders equal to the spare stock level s . The objective for establishing adequate spares levels is to compute the steady-state probability distribution for the number of items in resupply. To do this, they generalize and extend Palm's theorem to include demands that have a compound

Poisson distribution. They provide a proof of this extension in an appendix to the article. In their work, they present two cases: the backorder case and the lost sales case. For this research, only the backorder case is relevant because aircraft spares in an isolated location would be sole-source (i.e. no secondary source for spares exists). They provide three measures of merit:

- 1) Ready Rate - the probability that a unit has no backorders when reviewed at random points in time.
- 2) Fills - the expected number of demands that can be filled immediately from stock during a fixed period of time.
- 3) Units in service - the expected number of items in the resupply pipeline at any given point in time.

They also provide algorithms and FORTRAN code to calculate these performance measures in the appendices to the article.

Sherbrooke (1966) wrote the Multi-Echelon Technique for Recoverable Item Control (METRIC) model. Originally, it had three purposes: 1) Optimization of base and depot stock levels subject to cost or performance constraints, 2) Redistribution of current stock levels between base and

depot locations to provide optimal system performance, and
3) Assessment of system performance based on current stock levels at each location against a selected scenario.

Sherbrooke also presents several general model assumptions that he says are not exactly true but provide good approximations to reality:

- 1) The actual system objective is to minimize the sum of backorders across all items at all bases for a weapon system.
- 2) The demand for each item is a logarithmic Poisson process.
- 3) Demand is stationary over the prediction period.
- 4) Decisions on where to repair an item depend only on the difficulty of the repair
- 5) Lateral resupply is not allowed.
- 6) No condemnations are considered.
- 7) The depot doesn't batch items for repair.
- 8) Allows different priorities by base and item but not within an item.
- 9) Demand data from different locations can be combined.

He provides rational support for each of these assumptions.

He goes on to provide some mathematical development and

theory for the model and suggests methods for collecting and analyzing input data. He closes by presenting several model applications. They include cost-effectiveness decisions, minimum stock levels, maximum stock levels, base repairable percentage estimation, and average base repair time estimation. Sherbrooke (1968b) also provides a non-technical management version of this report and a shorter technical version (Sherbrooke, 1968a) in Operations Research. He discussed the METRIC evaluation criteria of minimizing expected base backorders subject to a budget constraint and its superiority to fill rate, the number of units supplied divided by the number demanded in Sherbrooke (1971). He points out these are still both measures of supply effectiveness and only indirectly operationally relevant. Therefore, Sherbrooke recommends using the number of aircraft Not Operational for Supply (NORS) as a more relevant operational measure of merit. The problem with using NORS, however, is that a model to minimize NORS aircraft is not mathematically tractable because the objective function is not a separable function of item performance measures. In addition to some of the METRIC mathematics, he also presents a case study of the F111 aircraft. He uses this case study to demonstrate the effectiveness of the model. First, he computes an F111

package of spares using the METRIC model subject to two budget constraints of 3.49 and 3.89 million dollars. Next, he evaluates these packages using NORS aircraft as the measure of merit. Then experimenting with other stocking policies, he tries to obtain spares packages that give better NORS values with the same budget constraints. After many trials, the best spares package performance in terms of NORS aircraft was less than 1 percent better than the METRIC model performance. He considers this reassuring but warns that the results are empirical and based on only one test sample. Also, additional policy trials may still reduce the NORS value although Sherbrooke does not think much improvement will occur. This becomes the first in a series of models that will eventually form the basis of the Air Force aircraft recoverable spares assessment and requirement computation system.

Muckstadt (1973) wrote the MOD-METRIC model which was an extension of the original work of Sherbrooke by allowing a hierarchical or indentured parts structure to be considered. Muckstadt (1976) continued this work and in 1976 he wrote the Consolidated Support Model (CSM). It was an extension of the MOD-METRIC model that allowed a three echelon system to be analyzed versus MOD-METRIC's two echelons. This model also added an additional constraint

or assumption. This constraint limits line replaceable units (LRUs) fixed at the intermediate repair facility to no more than one broken shop replaceable unit (SRU) per LRU. The model allowed users to look at requirements for not only assemblies or line replaceable units (LRU) but also for the subassemblies or shop replaceable units (SRU). His article goes on to describe the mathematical development of this extension and provides an algorithm for determining stock levels. He then uses this algorithm to compute stock levels for jet engines at base and depot level. Muckstadt points out that although he has included only two levels of indenture, the analysis method is easily extended to additional levels of indenture.

Finally, Hillestad and Carrillo (1980) made a significant breakthrough by finding a method to model nonstationary demand and service rates. This was important because assuming stationary demand patterns when transitioning from peacetime to wartime activity was not considered reasonable. This particular work is very mathematical and theoretical. Here they were able to produce many time dependent measures of system performance and derive transient results for nonstationary distribution periods. These results laid the groundwork for the Dynamic Multi-Echelon Technique for Recoverable Item Control (Dyna-

METRIC) model. Hillestad (1982) then wrote the Dyna-METRIC manual which provides a less rigorous mathematical description of the model along with a description of its capabilities and basic assumptions. Deming and Hobbs (1983) provide a brief but functional comparison of Dyna-METRIC and MOD-METRIC showing that they have similar theoretical foundations but Dyna-METRIC has several features that MOD-METRIC does not. It can consider three supply echelons versus only two in MOD-METRIC. Dyna-METRIC can also estimate ready rates, sortie rates, and other measures of aircraft readiness. The only cost for this expanded capability is Dyna-METRIC's inability to optimize system-wide spares levels which MOD-METRIC can do. Dyna-METRIC can only optimize at the base level.

According to Pyles (1984), Dyna-METRIC was developed to provide five new pieces of logistics information for decision makers. This information includes:

- 1) Operational performance measures.
- 2) Effects of wartime dynamics.
- 3) Effects of repair capacity and priority repair.
- 4) Problem detection and diagnosis.
- 5) Logistics performance assessments or spares requirements determination.

He also gives examples of the typical Air Force performance measures for this time period which include:

- 1) Resource Counts (ex. shelf stock and War Reserve Material (WRM)) such as on-hand versus authorized percentages
- 2) Process delay times (ex. repair times and order and ship times)
- 3) Peacetime customer satisfaction proxies (ex. percentage of requisitions filled, percentage of aircraft Not Mission Capable (NMC), and cannibalization rates)

Then he points out that although each of these measures provides some measure of overall support for U.S. Air Force systems, there is no integrated method to assess overall logistics support or even the relative importance of individual components.

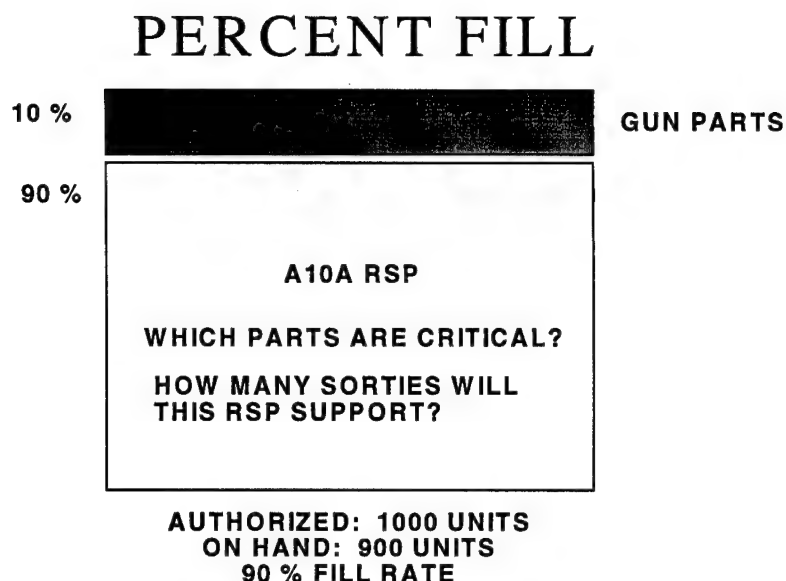


Figure 8: Fallacy of percent fill

As one can see from the figure above, the set of spares for this aircraft, the A-10, is 90% complete which might lead one to the false conclusion that this is a fairly capable package of spares. This would be a false impression because the primary mission of the A-10 aircraft is killing tanks using a 30 MM gatling gun. It turns out that the 10% of the spare parts missing are all gun parts. As such, as soon as the gun broke on the aircraft, there would be no replacement parts and the A-10 would become nothing more than a flying target.

This, Pyles claims, is where Dyna-METRIC can provide support. It ties the resource counts and process delay times to the important operational concerns of number of sorties flown and number of fully mission capable (FMC) aircraft available using dynamic wartime scenarios. A sortie flown is defined as one aircraft taking off, flying a predefined task, and landing safely while a FMC aircraft is an aircraft that is currently capable of accomplishing all of its wartime taskings without any restrictions. The report also provides an extended example of how to use the model for a single F16 squadron deploying to a single base. Pyles then provides a list of model limitations, a description of each limitation, and its impact on the modeling results. The limitations include:

- 1) Repair procedures and productivity that are unconstrained and stationary.
- 2) Forecast sortie rates that do not reflect flight-line resources and the daily employment plan.
- 3) Component failure rates that vary only with the user-defined flying program.
- 4) The assumption that aircraft at each base are basically the same.
- 5) The assumption that repair decisions and actions happen only after component testing.

- 6) Item failure rates that are not adjusted based on previous sorties flown.
- 7) The restriction that the repair processes are the same at every echelon's repair facility.

Gage and Ogan (1983) give some additional limitations that are also important. For example, the actual sorties flown by the model can never exceed the requested sorties. Also, the Not-Mission Capable due to Supply (NMCS) figure given by the model is not always equivalent to the number of grounded aircraft. This NMCS figure N simply means that the N aircraft are not able to accomplish all of their required missions. Put simply, the model does not consider Partially Mission Capable (PMC) aircraft or aircraft able to complete some but not all of their missions. The model still carries the METRIC assumptions that the repair and demand processes are independent, the depot facility is an unconstrained source of supply, and the intermediate repair facility distributes parts based on base cumulative flying hours. Gage and Ogan see Dyna-METRIC as a valuable management tool but warn that it's not the final truth. Because of the model assumptions and limitations, they recommend continual review of the outputs for validity and reasonableness.

The next step in the evolution of the multi-echelon, multi-indenture modeling effort came in the form of VARI-METRIC (Sherbrooke, 1986). Slay (1980) developed VARI-METRIC and Graves improved upon it. Graves (1985) showed that in 11% of the cases, METRIC computed stock levels that varied by at least one unit from the optimal while VARI-METRIC only differed in 1% of the cases. This improvement is achieved by taking into account not only the mean pipeline value as the previous models did but also looking at the variance of the pipeline quantities. Sherbrooke (1986) shows that in the case of multi-indenture, multi-echelon systems, VARI-METRIC gives estimates of backorder quantities that are very close to those calculated by simulation models. Thus, he believes that VARI-METRIC is an improvement over MOD-METRIC and Logistic Management Institute's Aircraft Availability Model. The Aircraft Availability Model is described by O'Malley (1983). To back up this claim, he presents several case study comparisons of the VARI-METRIC and MOD-METRIC models. Since that time, several investigators have developed exact solutions for this problem but they require more restrictive assumptions and substantially more computer time.

Dyna-METRIC version 3.04 became the standard aircraft unit assessment system for the Air Force in 1988. It fell under the umbrella of the Weapon System Management Information System (WSMIS) of the Air Force Logistics Command (AFLC) now known as the Air Force Material Command (AFMC) (Isaacson et. al., 1988).

This model represented a full scale implementation of the model discussed in Hillestad and Carrillo (1980), Hillestad (1982), and Pyles (1984). Version 4 is the version currently being used in the Air Force for weapon system assessment (Isaacson et. al., 1988). It was developed to assess worldwide logistics support of aircraft spares including the depot-theater interactions. This allows someone to assess movement of spares from one theater to another or to assess the impact of base/theater/depot repair processes, stock levels, transportation processes, cannibalization policies, and wartime plans on the military's combat capability. It also allows the user to view how spares support for subcomponents may impact combat capability by showing the impact of these parts in the repair process at each echelon.

DYNA-METRIC LOGIC EXAMPLE

1. COMPUTE PIPELINE QUANTITIES (DEMANDS-REPAIR)
 DEMANDS: 15
 REPAIR: 5
 PIPELINE: 10
 2. COMPUTE BACKORDERS (STOCK BALANCE - PIPELINE QUANTITY)
 STOCK BALANCE: 1
 PIPELINE: 10
 BACKORDERS: (9)
 3. COMPUTE TNMCS AIRCRAFT (CONSIDERING QPA & CANNs)
 QPA: 1
 CANNs: 7 FROM OTHER GROUNDED A/C
 TNMCS: 2 FOR THIS SPECIFIC ITEM
 4. COMPUTE SORTIES
 (PAA - TNMCS EQUAL FMC ACFT)
 (FMC ACFT X SGP EQUAL SORTIES)
- | | DAY 5 | DAY 20 |
|-----------------|----------------|----------------|
| FMC ACFT | <u>24-9=15</u> | <u>24-15=9</u> |
| SORTIE TASKING: | 45 // 3.00 | 36 // 4.00 |
| MAX PER ACFT: | 3.50 | 3.50 |
| SORTIES: | 15*3.00 (100%) | 9*3.50 (87.5%) |
5. IDENTIFY COMPONENTS WHICH PREVENT MEETING FMC GOAL

Figure 9: Dyna-METRIC computation flow
(DMAS Training Guide, 1994)

The figure above gives a simple overview of the computations that occur in the Dyna-METRIC model to assess sorties flown, FMC aircraft, and a prioritized list of the expected problem parts. Pipeline quantities for each item on the jet are computed and then, based on the current on-hand stock balances, backorders are computed for each item. From these backorders, the Total Non-Mission Capable Supply (TNMCS) or non-FMC aircraft can be computed. Knowing the number of grounded jets, the number of FMC aircraft can be calculated. Now the total number of sorties flown can be

calculated based on the FMC aircraft and the maximum sorties per aircraft. The problem parts are listed in descending order based on the number of aircraft the part is grounding.

Isaacson, Boren, Tsai, and Pyles (1988) give a detailed description of the differences between version 3.04 and version 4 which include:

- One more level of indenture
- One more echelon of repair and resupply
- Capability to analyze more than one aircraft-type at a time
- Additional detail in pipeline descriptions
- Additional output capabilities

In 1987 Slay and King (1987) developed the Aircraft Sustainability Model (ASM) which incorporates the VARI-METRIC theory to improve the computation of aircraft spares. Because of the improvements, it has been accepted as the Air Force standard for computing MRSP spares levels. It allows the user to give two different availability goals.

As mentioned earlier, one assumption of the METRIC-type models is unconstrained repair facilities. In 1989

Tsai (1989) wrote the Dynamic Simulation of Constrained Repair (Dyna-SCORE) to try to study the impact of maintenance policies on weapon system availabilities. The model's output includes summaries of job processing times, component pipeline contents, backorder quantities, weapon system availabilities, and equipment utilizations. The problem with Dyna-SCORE is that it is a single-echelon model that focuses on only one repair shop at a time and assumes that all other shops and echelons have no impact on aircraft availability except to fill requisitions or create demands for the modeled shop (Tsai, 1989). Dyna-METRIC version 5 (Isaacson and Boren, 1988) was an attempt to extend the model to consider constraining or controlling repair processes and uncertainties. It attempted to do this by replacing the pipeline component calculations based on Palm's Theorem with a Monte Carlo simulation. In this attempt to look at constrained repair, the model's biggest limitation is that it does not model a LRU's subcomponents or repair parts. This implies that a part will never be delayed in a repair facility while it waits for repair parts (SRUs) (Isaacson and Boren, 1988). Version 6 released in 1993 is an enhancement of version 5. It was an attempt to place back in the model the version 4 features that had been left out of version 5. It still uses Monte

Carlo simulation for determining pipeline quantities and attempts to fix the version 4 problem of computing pipeline demands based on scheduled flying hours rather than actual flying hours. Its most significant limitations include an inability to compute spares requirements and very long run times (Isaacson and Boren, 1993). Two other versions of the model which also exist should be mentioned. Dyna-METRIC Microcomputer Analysis System (DMAS) is a microcomputer version of the Dyna-METRIC version 4 model used by the Air Force. It has restricted capabilities and is primarily intended for base-level users who are doing unit level requirements and assessment calculations (DRC, 1993a). Major Command Dyna-METRIC Microcomputer Analysis System (MAJCOM-DMAS) is a Windows-based microcomputer version of Dyna-METRIC version 4 but is intended for more sophisticated users and scenarios. It provides access to a much larger selection of the version 4 capabilities. This package has all of the features of DMAS but also allows multi-unit, multi-echelon, theater-level assessments (DRC, 1993b).

Having studied the background of the current modeling efforts in this area, it is also important that we look at the importance of a timely and responsive tool to solve problems in this area.

2.4 Timeliness

2.4.1 Timeliness Background

The issue of timeliness of response/results is not unique to this research. Most military combat decisions are made in a time constrained environment. Timeliness in the civilian world is just as critical. Just-in-time manufacturing is built on providing the right things in the right places at the right time (Blackburn, 1991a). Unresponsive information systems or decision support tools are useless in these environments. Time-based competition is another area where the importance of time is stressed (Blackburn, 1991b). Timeliness and accuracy of information in support of product development decisions can mean the difference between successful product development and bankruptcy. Common sense says that unless the analysis tool can provide reasonable answers in a timely manner, the tool is useless.

Surprisingly, very little literature exists that discusses the impact of decision support aides on decision timeliness. Bittel (1991) in his book on time management discusses the usefulness of management information systems

to decision making but he is like most of the other sources on this subject, which assume if the system exists it is capable of delivering the information in a timely fashion. Sankar (1991) is another example of this approach.

Generally, the only concern expressed in these books is the potential for information overload. This is contrasted with Sprague and Carlson's (1992) comment that poor response times have been one of the primary causes of the mismatch between DSS capabilities and decision-maker's requirements. Another common theme in the literature is the need to implement a decision immediately because delay tended to result in evasiveness and procrastination (Bittel, 1991). Lee, Moore and Taylor (1981) and Hillier and Lieberman (1986) also emphasize the importance of the implementation phase in the problem solving process.

Thierauf (1975) stresses that the goal of building a decision support system (DSS) is to provide accurate and timely information. Thierauf (1988) also says that there is a trade-off between operational efficiency and effectiveness. He claims that improving operational efficiency leads to faster decisions, better turnaround time, and reduced costs while emphasizing operational effectiveness leads to better decisions. Sage (1986) promotes the need to consider the decision time horizon

when designing a system and Rouse (1986) implies the need to test this time horizon as part of operational testing. Rivett (1980) approaches timeliness from a slightly different angle. He stresses the need to consider the timeliness of a model development project and warns that if too much time is spent developing a tool, the need or interest in the project will be gone by the time it is finished.

The only article that dealt directly with the issue of model timeliness was by Coll, Coll and Rein (1991). It studied the impact of a DSS on the ability of an individual to find the best solution to a management problem. The study broke participants into three groups. The first group was permitted to use a DSS, the second was given a formal written, structured method to follow, while the third group was given no formal method. The results were surprising. The group using the DSS took the longest of the three groups to make decisions, were the least satisfied with their method, and felt they used the least creativity in finding a solution. They also finished second to the unstructured group in finding the "best" solution. The unstructured group also completed the problem-solving task in the shortest time of the three groups. I should point out that the study only looked at

one problem and one DSS so results certainly could vary depending on the type of problem and the information available to the participants but it is interesting that the automated tool did not necessarily improve the decision-making process.

Simulation often has this problem because of the need to make multiple runs to achieve acceptable confidence in the model's results. In addition to the model's restrictive assumptions, run times were another major factor leading to the lack of acceptance of Dyna-METRIC version 5 which included simulation elements in the model. Isaacson and Boren (1993) admit that adding simulation elements to their models has significantly increased run times.

Failure to provide timely responses can occur for a number of reasons. First, the data requirements for a particular method might not be available or obtainable in the time frame required. Second, even if the data is available, if the method requires a mainframe computer or extremely expensive software support to maintain, it is not likely to be available or gain acceptance. This is one of the primary reasons DMAS was developed. Prior to this development, the only way to run the Dyna-METRIC model was

by using a mainframe computer which is not accessible to the average combat unit.

Finally, the question of runtime or, more aptly, time-to-results must be addressed. Several issues come to mind. First and foremost is, how long does it take for an analyst to go from tasking to desired results? If he can not do it in the decision time available using the tool, the tool is not very useful.

The actual tool usage time can be broken into two parts. The first time component is the human interaction time with the model or the care and feeding of the model. This is the time someone must actually be involved with the model in order to get results. The second time component is any computer processing time that can take place without human intervention. If the model cannot perform the stand-alone processing in the time frame required, it is clearly a failure. When DMAS was first released, a combination of inefficient program coding and slow PCs (Z-150s) resulted in processing times that could exceed 18 hours. This is clearly unacceptable, especially if the analyst makes a data entry error or wants to perform a what-if analysis. Any complete analysis with these tools could take days which, as discussed earlier, is not reasonable.

But even if the processing time can be done within the required time frame, the model may still be a failure if the it requires a great deal of human intervention to properly operate. It is likely in a time-sensitive environment that the analyst will have more than this task to perform. This is certainly true in my application where the unit may be getting ready to go to war and running this model will not likely be a high priority for the supply sergeant (Sebring, 1994).

2.4.2 Military Timeliness Issues

Timeliness was a critical issue in Desert Shield/Desert Storm. Current deployment plans give front-line units 72 hours or less to move to their forward location. This does not allow much time to identify spares shortfalls and fill them. The sequence of events at a deploying unit would occur similar to this. When the deployment order was received, the tasked unit or units would identify the personnel and assets to be moved. In the case of aircraft spares, the Mission Readiness Spares Package (MRSP) would be deployed.

To perform a MRSP assessment under the current system, several pieces of information and equipment are necessary: a PC computer with the Dyna-METRIC Microcomputer Analysis

System (DMAS) loaded, a DMAS file with the unit's MRSP failure, repair, and authorized stock data, and the current on-hand, in-repair, and backorder asset positions for the unit. Gathering all of this information in 72 hours would be difficult, but conveniently, each unit should have all the equipment and data on-hand except the current stock positions. Policy recommends this information be on-hand but does not require it (AFR 67-1, 1993). The unit can get the stock information by running an R26 DMAS report extract (AFR 67-1, 1993). Sebring (1994) claims that running the R26 can take from 3 to 4 hours depending on the other competing demands for the Standard Base Supply System computer.

Another constraint on the deployment timeline is the need to have all supplies packaged and ready for aircraft loading 24 hours after movement notification. If shortfalls are identified in the assessment process, they must be located at other sources manually or by using the Multi-Asset Sourcing System (MASS). This can be a time-consuming process taking several hours especially if the unit is not the only one deploying (i.e. multiple sources competing for the same limited resources). Even if the item can be found, it then has to be delivered in time to be palletized and placed on the airlift aircraft. As one

can see, the less time it takes to identify the spares shortfalls, the better prepared the unit will be especially since this is only one of many tasks the unit needs to perform before a deployment.

Related to the timeliness issue is user-friendliness. The people who would likely be using this system are supply and warehouse managers and are generally not analytically or computer-oriented. If the system is not easy to use and, in addition, does not provide quick, accurate results, supply managers will probably ignore it. Early versions of DMAS, which were slow and inefficient, suffered from this problem.

2.5 Verification And Validation: A Description

Model verification and validation are important aspects of any model building effort. Many definitions of these terms exist in the literature. Pritsker (1986) defines verification as "The process of establishing that the computer program executes as intended." Khoshnevis (1994) defines it as a "computer implementation of the model that is error-free," and he points out that verification is not concerned with establishing whether the model is reasonable. Pegden (1990) agrees that it is

checking the model to see that the model performs as expected and intended. Law and Kelton (1991) see verification as a debugging process.

Validation is defined by Pegden (1990) as the process of raising the user's confidence to an acceptable level such that he is willing to use inferences drawn from the model to impact the real system. Shannon (1975) agrees with this definition. Pritsker (1986) calls validation the process of achieving a desired accuracy between the model and the real system. Khoshnevis (1994) defines it as the process which establishes that the model and the input data represent the important aspects of the modeled system. Gross and Harris (1985) and Lee, Moore, and Taylor (1990) view validation as a combination of the verification and validation processes while Shannon (1975) credits Fishman and Kiviat (1967) with breaking the model evaluation process into three steps:

- 1) Verification - the model runs as the modeler intended,
- 2) Validation - there is agreement between the behavior of the model and the real system,
- 3) Problem Analysis - the model user is to draw significant inferences from the model results.

The question of verification and validation boils down to answering the question, does the model adequately represent reality? Specht (1968) warns that the model is just an "analog of reality" and may not represent every aspect of reality. He stresses that the critical thing is that the model outputs answer our questions in an appropriate and valid manner. Because we do not have complete knowledge, he states that the best we can hope for is the ability to answer the following questions:

- 1) Does the model clearly and correctly describe the known facts and circumstances?
- 2) When the input parameters are varied, do the results remain consistent and reasonable?
- 3) How does the model handle special cases where we have some indication of the outcome?
- 4) Can it assign causes to known effects (Specht, 1968)?

Shannon (1975) states that it is impossible to prove that any model is the true or correct model of a system but adds that this is seldom important since we are primarily concerned with validating the insights gained from the

model. He stresses that these insights provide the model utility not the model structure accuracy. Specht (1968) agrees. He says we should not be upset that a model does not look like the real thing or that it does not represent all of reality. He adds that several models of the same reality may be valid depending on the questions asked and the decisions affected. Pegden (1990) clarifies this discussion by saying that verification is where the modeler gains confidence in the model and validation is where the modeler transfers this confidence to others.

As stated above, validation is the process of convincing the decision maker or model user that the model accurately reflects the real system for the user's decision making or analysis purposes. Pritsker (1986), Gross and Harris (1985), and Lee, Moore, and Taylor (1990) all agree that validation is the more difficult of the two evaluation tasks. The ultimate goal of any model is to aid the decision maker, therefore, the modeler would always like to test the correctness and relevance of the results against reality. Since this is not always possible, says Specht (1968), the best we can sometimes hope for is to be honest. Further, claims Quade (1968), no matter how hard the analyst strives to maintain a scientific inquiry or to follow scientific methods, military systems analysis is not

an exact science. Although it may appear rational, analytical, and objective, do not be fooled adds Quade.

Human judgment is used in the analysis for:

- 1) creating the analysis design,
- 2) determining the relevant factors,
- 3) determining the interactions to model and which interactions to leave out,
- 4) choosing the alternatives to consider,
- 5) selecting input data,
- 6) analyzing and interpreting results.

Thus, Quade cautions, judgment and intuition are fallible and, therefore, caution is advised for both the modeler and the decision maker to avoid biasing the model results.

Specht gives a similar warning:

This fact - that judgment and intuition and guesswork are embedded in a model - should be remembered when we examine the results that come, with high precision, from a model. (Specht, 1968)

Verification is usually performed using a manual check of the calculations, claims Pritsker (1986). Gross and Harris (1985) recommend a similar method. Pegden (1990) says the best support for verification comes from proper program design, plus clarity, style, and ease of

understanding. He also recommends a four step approach to verification that includes establishing a skeptical frame of mind, using outside skeptics, conducting model and experimental walk-throughs, and performing test runs. As part of this process, Law and Kelton (1991), Khoshnevis (1994), and Pegden (1990) all recommend computer animation as a valuable verification tool, if it is available. Khoshnevis (1994) states that models usually fail to operate correctly as a result of coding errors or logic errors. Coding errors, he says, are usually the easier of the two types to find because they stop the program's execution process and are usually located by the compiler's error checking system. Logic errors, on the other hand, are much more difficult to find and correct. Both Khoshnevis (1994) and Pegden (1990) provide lists of the most common error types. These lists include things such as data errors, entity flow problems or deadlocks, problems with units of measurement, and overwriting model variables. They both also suggest techniques to avoid these common errors. Law and Kelton (1991) provide eight different techniques for verification that include modular code construction and testing, debugging using a program trace function, animation, and structured walk-throughs.

Specht (1968) and Law and Kelton (1991) each present equivalent three step approaches to model validation. They recommend:

- 1) testing the face validity of the model - These tests determine whether a model seems reasonable to experts familiar with the system under study.
- 2) testing the model assumptions - This involves testing quantitatively the assumptions made early in the model development.
- 3) testing the reasonableness of the input-output transformation - This involves testing whether or not the model produces results similar to the real or proposed system.

Steps two and three above can use a number of statistical techniques and methods for accomplishing these tasks. These techniques include statistical tests of means and variances, regression, analysis of variance, autocorrelation, and non-parametric tests. Shannon (1975) warns, however, that each of these test procedures comes with its own set of assumptions which must be considered. Hillier and Lieberman (1986), Pritsker (1986), Gross and Harris (1985) and Pegden (1990) all agree that using

standard statistical tests is the way to go if data exists for comparison. If data does not exist, Hillier and Lieberman (1986) recommend using face value testing but they also suggest trying to collect system data from field testing, if possible. Then use experts in the field to carefully select test scenarios and perform sensitivity analysis on the model inputs and outputs. They add, however, that field testing is frequently costly and time-consuming and therefore, often impractical. Regardless, they emphasize the importance of convincing the decision-maker of the credibility of the model for decision-making purposes. Care must be taken when using past performance as the "TRUE" system value. Pritsker (1986) warns that past performance may only represent one sample and not necessarily the exact answer. Dalkey (1968) agrees and adds that even if the two disagree, the model may still be valid. He points out that chance, model detail, and a commander's decisions all affect historical outcomes. He emphasizes that a major factor in war is a commander's decisions and presently, there is no adequate way to model these decisions.

Lee, Moore and Taylor (1990) warn that simulation modeling is particularly susceptible to the garbage in-

garbage out syndrome. Pritsker (1986) sees the validation process as a way of answering two questions. They are:

- 1) what is the inherent variability within the model,
and
- 2) what can be inferred about the real system
performance from the model performance?

He claims the first question deals with understanding the model and assuring the model operates as intended. The second deals with the model's usefulness. The first involves obtaining a detailed statistical analysis on the precision and sensitivity of the model. The second question is related to the modeled system and therefore, model dependent, and as such there are no general analysis methods that can be recommended beyond the standard statistical tools.

Khoshnevis (1994) states that there are two different approaches to model validity. Empiricism and Rationalism. Pegden (1990) adds a third he calls Positive Economics. Rationalism is an approach that assumes most of the underlying assumptions a model is based on are obviously true and therefore in no need of proof. Logical deductions are then used to develop the model and, as such, if the

assumptions and logic are valid, then the model is valid. The Empirical approach demands that every assumption and result be empirically tested and validated. No assumption is allowed that cannot be independently tested or verified. The Positive Economics approach only requires that the model be capable of predicting the future. It is not concerned with validating the underlying model assumptions or structure. Thus, if the model has good predictive capability, it is assumed to be valid. Although Law and Kelton (1991) and Khoshnevis (1994) both provided several approaches to increasing model validity, Pegden (1990) provided the most complete list. He broke these test approaches into three areas. They include tests for reasonableness, tests of model structure and data, and tests of model behavior. Within tests for reasonableness, he recommends checking the following:

Continuity: Small changes in the input data should result in small but appropriate changes in the decision variables.

Consistency: Similar runs of the model should result in similar outcomes.

Degeneracy: If model features are removed, the decision variables should reflect their removal.

Absurd Conditions: This has two parts.

(1) If the modeler provides unusual data input, the model should not give unusual or unpredictable results.

(2) The model should never generate absurd or impossible situations.

For testing the model structure and data, Pegden (1990) proposes these tests:

Face validity: This is accomplished by asking experts of the modeled system whether the model's behavior seems reasonable.

Parameters and Relationships: This area includes tests of assumptions concerning parameter values and variable relationships and typically involves statistical tests such as tests of means and variances, regression analysis, and goodness of fit tests.

Structural and Boundary Verification: The key here is to ensure that the structure of the model does not clearly contradict the real system.

Sensitivity Analysis: This is done by varying the model input parameters and checking the impact of

these changes to the model's outputs. This should give us some idea of how sensitive the model is to small changes in the input parameters.

Also, Pegden (1990) suggests a number of tests for investigating model behavior:

Behavior Comparison: This involves comparing the model output to the real system results. He lists a number of statistical tests available for this testing including the Chi-Square test, Kolmogorov-Smirnov test, and regression analysis.

Symptom Generation: These tests take different forms but answer questions like:

- Can the model produce the same difficulties that show up in the real world?
- Does it produce the same results after a change as the real system did after similar changes?

Behavior Anomaly: If the model gives results that conflict with the modeler's expectations, can the modeler find examples of this

behavior in the actual system. If he cannot, there may be a problem with the model.

Behavior Prediction: Use the model to predict system performance during field tests by controlling inputs to both the model and the actual system.

Finally, Law and Kelton (1991), Shannon (1975), and Pegden (1990) stress that verification and validation are ongoing tasks and should not be delayed until the end of the project, to be performed if time and money permit. Figure 10 illustrates the continuous roles verification and validation play in the model development process.

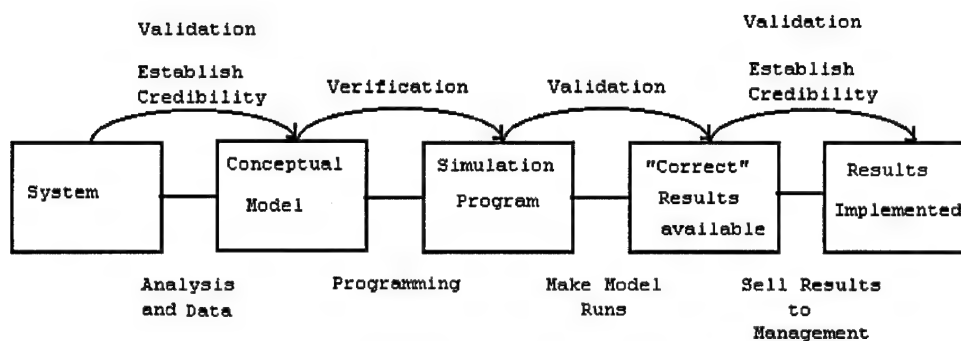


Figure 10: The continuous nature of verification and validation (adapted from Law and Kelton (1991))

This section concludes with a thought by Shannon (1975):

The question of validation is thus two-faced: determining that the model behaves in the same fashion as the real system; validating that inferences drawn from the experiments with the model are valid or correct. In concept, both these points resolve themselves to the standard decision problem of balancing the cost of each action against the value of the increased information and the consequences of erroneous conclusions.

2.6 Summary

This section covered background material necessary for the understanding of the research problem. First, general inventory measures and models were discussed. Then the role of queueing models were discussed as a solution technique in this area. The METRIC models are the models currently in practical use but they have some problems when applied in finite source queues or when sampling without replacement is important. Timeliness of response is also a critical aspect of this research and is discussed in this chapter. General background material on verification and validation is also provided.

CHAPTER 3

METHODOLOGY

The research methodology can be broken into three general areas. These areas include data gathering, model development and validation, and model output analysis. This research identifies the current shortcomings of the Air Force combat aircraft assessment techniques which include the use of the Dyna-METRIC model. It then considers several analytic techniques to produce a better assessment method. The approach in each area is discussed below.

3.1 Data Gathering

Data gathering or more accurately information gathering occurred in several ways. Aircraft demand and repair information came from HQ ACC/LGSW although HQ AFMC was a secondary source. Part of this research also considered what data elements are actually necessary for these assessments. Current information on the model capabilities came from a wide variety of sources. In addition to the two sources listed above, information on

the model also came from HQ AF/LEYS, Logistics Management Institute (LMI), Dynamics Research Corporation (DRC) and RAND Corporation. Each of these groups has an active interest and involvement in the current model and future assessment developments. Military regulations and manuals also provided guidance. Finally, current literature was an additional source of information on the current assessment methods. Various models for comparison with the research model came from RAND, DRC, and HQ ACC/LGSW. The data gathering efforts feed the methodology development and data analysis section of this work.

3.2 Model Development

Methodology development and validation represent the heart of this research project. First, the current model and other related modeling techniques are discussed. Then the research method is covered.

3.2.1 Current Model

This research developed a clear understanding, explanation, and demonstration of the problems and shortfalls of the current assessment methodology. This was done through a thorough literature review and discussions

with the practitioners in the field (see section 2.3).

Based on this process understanding, other possible methods and modeling techniques were investigated as improvements to the current process.

3.2.2 Alternate Modeling Approaches

As part of this investigative process, the author also looked at other potential methods that were already available in the literature. These approaches provided fertile ground as possible starting points for contributions by the author. Several approaches appeared promising as starting points for this research. The first was based on Jacksonian networks (Jackson, 1957 and Jackson, 1963). Gross et al. (1983) used this networking method to develop a technique to optimize sparing levels and repair channels. It is an extension of earlier work done by Mirasol (1964). Mirasol built a model which considered a single-echelon, single-repair shop, infinite calling population, finite repair capacity system. Gross added to this work by considering a finite population of operating times and a two-echelon repair system. The drawbacks with this approach are that it only considers steady-state conditions and was built to optimize sparing levels based on cost and not to determine availability

based on sparing levels. This modeling approach was also discussed in section 2.2. Using a reformulated network approach gave promising results. It showed an order of magnitude improvement in the accuracy of the results over the current method for at least the sample scenario. Another promising approach used a method developed by Sherbrooke (1992) for the NASA space station Freedom. It was conceptually similar to the METRIC-type models but did not use Palm's theorem because it did not assume that the shapes of the repair distributions are unimportant. It also did not use an infinite operating population assumption to distribute backorders. One drawback of this method was that it still assumes an infinite calling population for demands. Kaplan(1989), however, developed a model similar to Isaacson (1988) but his model included the finite calling population assumption. This technique could possibly be adapted into Sherbrooke's model (1992) or possibly extended as a primary method.

This is a brief discussion of Sherbrooke (1992) and Kaplan's (1989) work. This study considered the advantages and disadvantages of both works and how they might be used in this research. Sherbrooke uses a convolution or sum of probabilities to describe the number of items due-in at the beginning of each resupply mission. It is:

$$\Pr\{DI = y\} = \sum_{x=0}^y p(x|m_1T)p(y-x|m_2T) \quad y = 0, 1, 2, \dots \quad (3-1)$$

where:

- y is the number of items due-in.
- m_1T and m_2T are the mean demands for items that require one resupply and two resupply cycles to repair, respectively.
- T is the time between resupply missions.

This is the expression he used to compute backorders on the space station:

$$\Pr\{BO = x\} = p(s_0 + x|mt) \Pr\{DI \leq s_g\} \quad (3-2)$$

$$+ p(s_0 + x - 1|mt) \Pr\{DI = s_g + 1\}$$

$$+ p(s_0 + x - 2|mt) \Pr\{DI = s_g + 2\}$$

$$+ \dots + p(0|mt) \Pr\{DI = s_0 + s_g + x\}$$

for $x = 1, 2, 3, \dots$

and where:

- s_0 is the stock level at the space station,
- s_g is the stock level on earth, and t is the time on station where $0 < t < T$.

For this research no periodic resupply is available but there is unit level repair so s_g could represent the items due-in from maintenance.

Sherbrooke handles redundancy using the hypergeometric distribution. The probability distribution for N systems, having x_1, x_2, \dots, x_N backorders, with a total of $y \equiv \sum_{j=1}^N x_j$ backorders across all N systems is:

$$\text{hyp}(x_1, x_2, \dots, x_N | y) \equiv \sum_{p=1}^n x_j = \frac{\binom{Z}{x_1} \binom{Z}{x_2} \dots \binom{Z}{x_N}}{\binom{NZ}{y}} \quad (3-3)$$

This is conditional on y because there must be y backorders among the NZ item locations. As a result the probability that systems $1, 2, \dots, i$ are up and $i+1, \dots, N$ systems are down, $S(i)$, is:

$$S(i) = \sum_{y=0}^{\infty} \Pr\{BO = y\} \sum_w \text{hyp}(x_1, x_2, \dots, x_N | y) \quad K \leq i \leq N \quad (3-4)$$

where W is the set of hypergeometric probabilities such that the first i systems are up and the last $N-i$ systems are down.

To help clarify this, consider an example taken from Sherbrooke (1992). Assume three systems are operating simultaneously. Each system consists of one item which is parallel redundant (i.e. $Z = 2$, $z = 1$, $N = 3$). The goal is to determine the probability that the first two systems are operating and the third system is not operating, $S(2)$. Since at least two backorders are needed to disable a system, the set W is empty for zero or one backorder ($y = 0$ or 1). When two backorders are present, $y = 2$, only one arrangement of these backorders will result in the third or last system being down and that combination is $x_1 = 0$, $x_2 = 0$, $x_3 = 2$. This gives the following hypergeometric probability:

$$\text{hyp}(0,0,2) = \frac{\binom{2}{0}\binom{2}{0}\binom{2}{2}}{\binom{3*2}{2}} = \frac{1}{15}$$

When three backorders occur, $y = 3$, there are several combinations that can be added to set W . They are $x_1 = 1$, $x_2 = 0$, $x_3 = 2$, and $x_1 = 0$, $x_2 = 1$, $x_3 = 2$. Each of these

combinations result in the first two systems up and the last one being down. This results in the hypergeometric probability:

$$\text{hyp}(1,0,2) + \text{hyp}(0,1,2) = \frac{\binom{2}{1}\binom{2}{0}\binom{2}{2}}{\binom{3*2}{3}} + \frac{\binom{2}{0}\binom{2}{1}\binom{2}{2}}{\binom{3*2}{3}} = \frac{1}{5}$$

When four backorders occur, $y = 4$, there is again only one combination that can be added to set W which allows the first two systems to be up and the last system to be down. It is $x_1 = 1$, $x_2 = 1$, $x_3 = 2$. This results in the hypergeometric probability:

$$\text{hyp}(1,1,2) = \frac{\binom{2}{1}\binom{2}{1}\binom{2}{2}}{\binom{3*2}{4}} = \frac{4}{15}$$

When five backorders occur, no distribution of backorders exists that allows the first two system to be up and the last system to be down. Using this set W of possible failure combinations to calculate the probability of the first two systems being up and the last one being down, $S(2)$, results in:

$$S(2) = \frac{1}{15} \Pr\{BO = 2\} + \frac{1}{5} \Pr\{BO = 3\} + \frac{4}{15} \Pr\{BO = 4\}$$

In the table below, this same logic is used to build the hypergeometric probabilities for all possible values of K.

Table 7: Examples of Hypergeometric Probabilities
(Taken From Sherbrooke, 1992)

Number of Back Orders, y	Probability That a Specific Configuration of Exactly I Systems of N will Operate			
	i = 3	i = 2	i = 1	i = 0
0	1	0	0	0
1	1	0	0	0
2	4/5	1/15	0	0
3	2/5	1/5	0	0
4	0	4/15	1/15	0
5	0	0	1/3	0
6	0	0	0	1

This may appear computationally burdensome but with the natural bounds on y, the hypergeometric probabilities are independent of the demand rate and stockage policy so

they only need to be computed once and then can be used over and over again for every system considered.

Next, Sherbrooke showed how this technique applies to groups of different systems where each type of system must be up for the entire group of systems to be up. Therefore, the probability that all N groups of systems are up is:

$$S(N) = S_1(N) S_2(N) \quad (3-5)$$

where $S_1(N)$ is the probability that all systems are up in the first type of systems while $S_2(N)$ is the probability that all systems are up in the second type of systems.

$S(N-1)$, below, is the probability that the first N-1 groups are up and system group N is down.

$$S(N-1) = S_1(N-1) S_2(N) + S_1(N) S_2(N-1) + S_1(N-1) S_2(N-1) \quad (3-6)$$

$S(N-2)$, below, is the probability that groups 1, 2, ... N-2 are up and groups N-1 and N are down.

$$S(N-2) = S_1(N-2) [S_2(N-2) + 2S_2(N-1) + S_2(N)] \quad (3-7)$$

$$+2S_1(N-1)[S_2(N-2)+S_2(N-1)]+S_1(N)S_2(N-2)$$

In this formula, the first factor of two ($2 \cdot S_1(N-2)S_2(N-1)$) appears because, if two systems are down for system 1 and one is down for system 2, there are two ways to select the location of broken system-type #2. The converse is true for the second factor of two ($2 \cdot S_1(N-1)S_2(N-2)$). The last factor of two ($2 \cdot S_1(N-1)S_2(N-1)$) occurs because if a specific system is down for system 1 and a different specific system is down for system 2, then there are two ways to choose these down systems that results in the last two groups being down. These equations get more complicated as the number of systems grows, but for this research, the number N is not large and the number $N-K$ that are allowed to be down is small.

Sherbrooke then presented a method to calculate the group availability. To get the availability of at least i of N operating groups, we must first calculate the probability of the first i groups operating and multiply this by the number of ways that i groups can be selected from the N groups. The availability is given below:

$$A(i) = 100 \left[S(N) + \binom{N}{N-1} S(N-1) + \dots + \binom{N}{i} S(i) \right], \quad K \leq i \leq N \quad (3-8)$$

As a starting point, this approach offered several advantages over other approaches considered thus far. It does not use Palm's theorem and is not restricted to steady-state conditions. Although it does not formally handle dynamic arrival rates, the problem could be segmented in such a way that this situation could be handled. For example, since we are not tied to a steady-state solution we can break up the solution into as many segments (days) as we want and then solve each piece separately and later combine them into a final solution. This method does not use the infinite population assumption to distribute the backorders among the failed systems which tends to overstate the unavailability (Sherbrooke, 1992) and it obviously handles component redundancies.

The method also has its disadvantages. They include the fact that the method assumes a infinite calling population. For the space station this was not considered a problem because of the high component reliabilities but this is not the case for the current problem. This approach also does not consider on-board repair or repair between the resupply periods. Everything is repaired on

the earth and returned to the space station on the next shuttle mission, if possible. It also only considers one level of component indenture and should be extended to two. A possible solution to the repair problem is to group systems by repair cycle times and use the repair cycle time as the shuttle resupply time for that group and compute the availabilities by repair time group. This would be somewhat crude but possibly effective.

Overcoming the infinite calling population assumption in this model is more difficult. Kaplan (1989) developed a model similar to Isaacson (1988) (i.e. Dyna-METRIC version 4) but his model included the finite calling population assumption. This technique could possibly be added to Sherbrooke's model or extended as a primary method. If Kaplan's method provided a clean way of calculating backorder probabilities (equation 3-2), it could be plugged directly into Sherbrooke's method to calculate redundant system availabilities with a finite calling population. Using Kaplan's method as a starting point would be difficult because it appears to suffer from many of the same assumptions and limitations that the current method does including using the infinite population assumption to distribute backorders. He also did not offer a way to

compute system availabilities using his method so this would need to be developed.

3.3 Research Model

The research model must overcome several challenges. These include the difficulties of a finite-calling population, item redundancy, and the proper distribution of component backorders. The figure below depicts the flow of parts in a typical aircraft logistics system. The following development presents the research model.

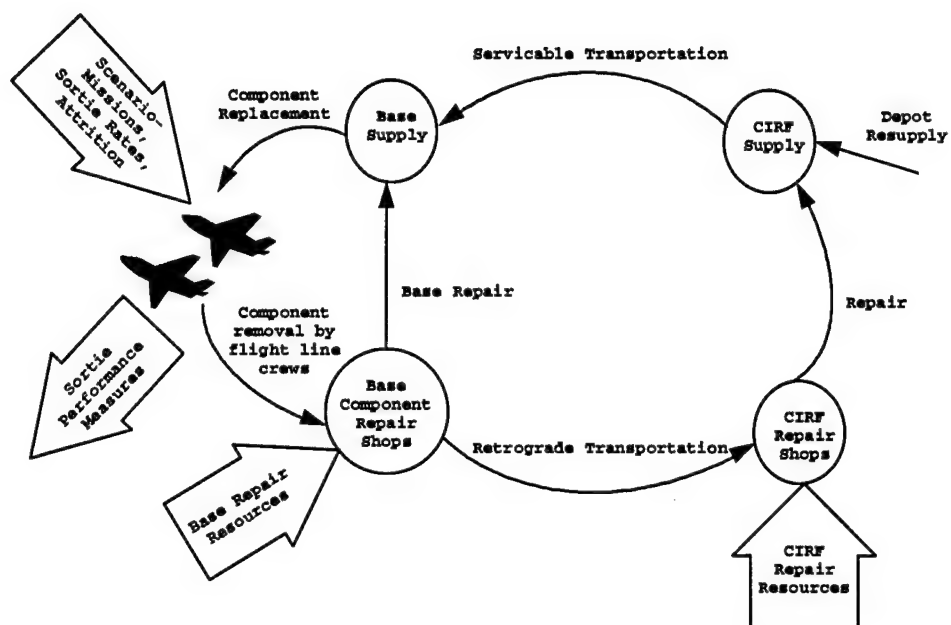


Figure 11. Typical aircraft logistics flow

3.3.1 Case I - One Part With S Spares

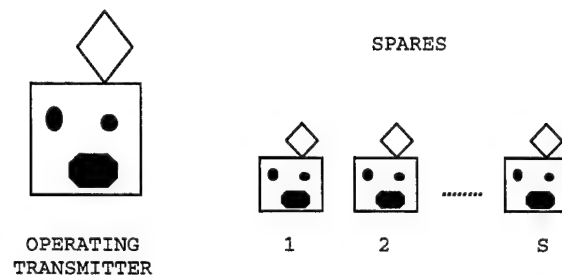


Figure 12. Case I - One transmitter and s spares

To introduce the research model, consider determining the probability distribution of failures for a radio transmitter. Assume the transmitter's failure distribution is exponential with a failure rate of λ . Also, assume the transmitter repair distribution has an exponential distribution with repair rate μ . For purposes of this discussion, f will equal the ratio of the failure rate and repair rate (λ/μ). Based on the general birth-death model

taken from Gross and Harris (1985), the following general formula is given for the probability of X failures, $P(X)$:

$$P(X) = P(0) \prod_{j=1}^m \frac{\lambda_{j-1}}{\mu_j} \quad (3-9)$$

They also provide an inductive proof of this formula. $P(0)$ can be calculated from the fact that for eq. 3-9 to be a valid probability distribution, $\sum_{X=0}^m P(X)$ must be equal to 1. Thus $P(0)$ can be thought of as a normalizing constant. Although this formula allows for state dependent arrival and service rates, λ_j and μ_j respectively, this work assumes λ_j and μ_j remain constant in all states for the period of interest.

To help illustrate the use of this and the other techniques presented in this section, let's consider a series of examples. For the first example, let's assume that only one transmitter is being used. The probability distribution of failures, $P(X)$, is simply:

$$P(X) = P(0) f^X / X! \quad X = 0, 1 \quad (3-10)$$

where X is the number of transmitter failures.

If s spares or replacement transmitters are allowed, the failure distribution becomes:

$$P(X) = P(0) f^X / X! \quad X = 0, 1, \dots, s+1 \quad (3-11)$$

Thus, if the failure rate for the transmitter were 0.4 failures per hour and the repair rate were 0.5 repairs per hour, then f would equal 0.8. If there were two spares in the system, the steady-state probability distribution of failures would be:

Table 8: Distribution of Failures

X	P(X)
0	0.4534
1	0.3628
2	0.1451
3	0.0387

These results imply, for example, that 14.51% of the time there will be two broken transmitters while 45.34% of the time there will be no broken transmitters. This distribution also tells us that for only 3.87% of the time there will be no transmitter operating. The next section

expands this case by allowing more than one component to operate at a time.

3.3.2 Case II - R Transmitters With S Spares

Consider Case II which will allow more than one transmitter to operate at a given time. Now assume that a communication system has a requirement to keep R transmitters operating at the same time. This situation is illustrated below.

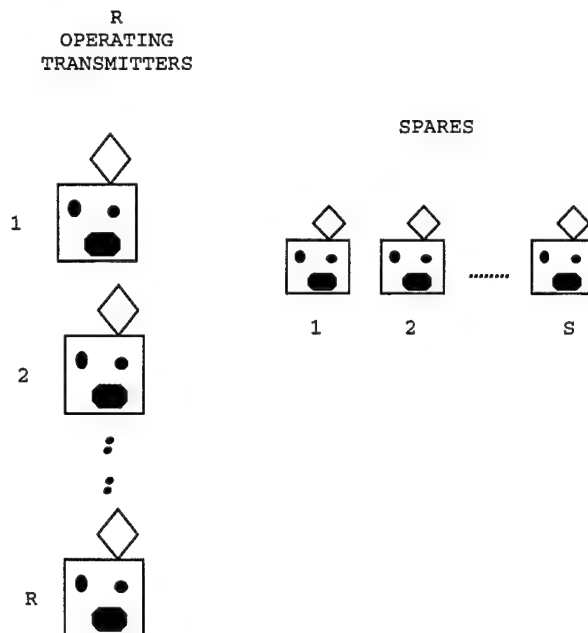


Figure 13. Case II - R operating transmitters and s spares

Eq 3-12 is a failure distribution function which accounts for the finite-calling population nature of this situation (i.e. the operating population consists of only the R transmitters and s spares). Once all the original operating components and spares are consumed, no more arrivals (failures) can occur. Also, as the number of operating components decreases, the arrival rate or failure rate naturally decreases.

Failure Distribution:

$$P(X = x) = \begin{cases} P(0) \frac{f^x R^x}{x!}, & x \leq s \\ P(0) \frac{f^x R^s}{x!} \frac{R!}{(R - x + s)!}, & s + 1 \leq x \leq R + s \end{cases} \quad (3-12)$$

where

x - Number of transmitter or component failures

f - Ratio of failure rate to repair rate (λ/μ)

P(0) - Normalizing constant to get the probability distribution to sum to one

s - Number of spare parts or components

R - Total number of transmitters or components being operated

The upper half of this equation is the same as the infinite calling population formula because there are enough spares to satisfy all demands due to failures (i.e. as $s \rightarrow \infty$, the

arrival rate becomes $(R\lambda)$, giving a $P(X = x) = P(0) \left(\frac{R\lambda}{\mu} \right)^x / x!$.

When, however, the failures exceed the number of spares available, the actual demand rate should be reduced to account for the reduced number of operating components. This is done in the bottom half of the equation by adding the $(R!/R^{x-s}(R-x+s)!)$ term.

Using the failure and repair information from the previous section's example, assume that three operating transmitters are needed (i.e. R is 3) and that there are three transmitter spares available ($s = 3$). The following failure distribution results:

Table 9: Case II Sample Results

X	P(X)
0	0.0955
1	0.2293
2	0.2751
3	0.2201
4	0.1321
5	0.0423
6	0.0056

The results of these calculations are interpreted in a manner similar to the last section. The probability of having three failed transmitters is 0.2201. The results of this table can also be combined to make statements about the three transmitter system. For example, since we desire three operating transmitters, the probability that all three transmitters will be operating is actually the sum of the probabilities that less than four transmitters will be broken or $P(0) + P(1) + P(2) + P(3)$ which equals 0.80. This means 80% of the time three transmitters will be operating. The next section expands the method to allow bundles of components to operate as a unit with possible component redundancy.

3.3.3 Case III - M Transmitter Units With Transmitter Or Component Redundancy

Now consider the situation where a transmitter unit i is actually made up of r_i transmitters and there are M transmitter units in operation (See Figure 14).

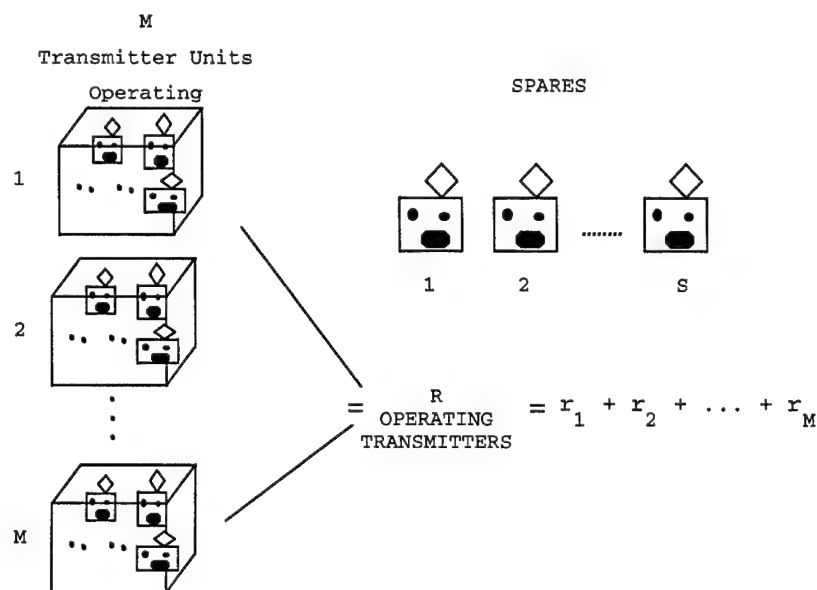


Figure 14. Case III - M transmitter units with s spares

This development implies that the total number of

transmitters in operation is $R = \sum_{i=1}^M r_i$. To properly

address the problem of transmitter redundancy, the hypergeometric distribution will be used to properly distribute all transmitter failures, X , across all of the transmitter units. The number of transmitter failures, y_i , on each transmitter unit i is such that $\sum_{i=1}^M y_i = x$. The distribution of these X failures across the M transmitters using the hypergeometric distribution is shown below.

$$\text{hyp}(y_1, y_2, y_3, \dots, y_M | X) = \frac{\binom{r_1}{y_1} \binom{r_2}{y_2} \dots \binom{r_M}{y_M}}{\binom{R}{X}} \quad (3-13)$$

where

M - Number of transmitter units being operated

X - Total number of transmitter component failures

r_i - Number of transmitter components on transmitter unit i

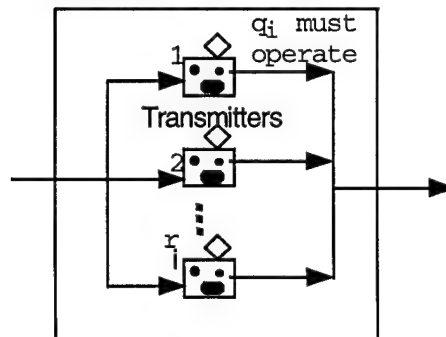
y_i - Number of transmitter component failures on unit i

such that $\sum_{i=1}^M y_i = x$

R - Total number of transmitters or components being operated

Redundancy within a transmitter unit i implies that not all the r_i transmitters must operate for the transmitter unit i to function properly. Therefore, assume that for transmitter unit i to operate properly only q_i of the r_i transmitters must be operating. Redundancy at the transmitter unit level must also be considered. This means that not all the transmitter units need to operate for the transmitter system to function properly. So, assume that I of the M transmitter units must be operating as a minimum. These two situations are illustrated in the figure below.

Redundancy
of
Transmitter Unit i



Redundancy
of
Transmitter System

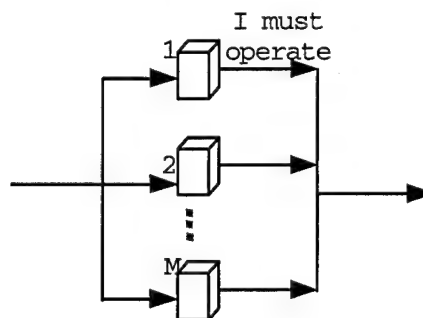


Figure 15. Transmitter redundancy

Now the failure distribution of the transmitter units can be calculated. To properly allocate the transmitter failures among the transmitter units, the probability $U(k)$ that the first $1, 2, \dots, k$ transmitter units are up and the $k+1, k+2, \dots, M$ units are down is computed using:

$$U(k) = \sum_{x=0}^{\infty} P\{X = x\} \sum_T \text{hyp}(y_1, y_2, \dots, y_M | x) \quad I \leq k \leq M \quad (3-14)$$

where

T - The set of transmitter component failure combinations that result in the first k transmitter units up and the rest down

I - Minimum number of transmitter units that must be available

This set T of possible failure combinations which will result in the first k transmitter units operating and the last M-k transmitter units being down, will also limit the number of hypergeometric computations that must be performed. For example, if each transmitter unit has two transmitters, $r = r_i = 2$, and only one must operate for the unit to function properly, then q_i is one. If four transmitter units exist ($M = 4$) and we are interested in the situation where the first three units are operating and the last unit is not operating ($k = 3$), then the following table contains the set T for this example.

Table 10: Set T for First Three Transmitters Up and the
Last One Down

Back Orders on Unit i				Total Back Orders for the System of Transmitters - X
Y ₁	Y ₂	Y ₃	Y ₄	
No Members				1
0	0	0	2	2
0	0	1	2	3
0	1	0	2	
1	0	0	2	
1	1	0	2	4
1	0	1	2	
0	1	1	2	
1	1	1	2	5
No Members				6+

This table was built by taking each value of the total system backorders and trying to allocate them among the transmitter units in such a way that the first three units are up and the last one is down. For example, when there is only one system backorder, there is no allocation scheme which will break the last transmitter unit since it has a transmitter redundancy. Thus, even if the backorder is on the last unit, the last unit will still be able to operate.

When there are two backorders, there is only one combination which will take down the last system and that is when both backorders are on the last transmitter unit. This logic proceeds down the chart until you have six system backorders, when no allocation of backorders across the system will allow only the last transmitter unit to be down. In addition, any higher number of system backorders will also not allow only the last transmitter to be down.

Building on our previous examples, assume that the system we are working with has three transmitter units which are made up of two transmitters apiece and that these transmitters are redundant. The system also has no transmitter spares. In terms of our notation, we have:

$$M = 3, r_i = r = 2, q = q_i = 1, s = 0, \text{ and } R = 6$$

Using this information and the results from the previous sections, we can find the probability of having the first K transmitter units up and the last $M - K$ units down. The results are presented in Table 11 below.

Table 11: Sample Probabilities of the First K Up and the
Rest Down

K	U(K)
0	0.0077
1	0.0313
2	0.1272
3	0.5168

These results imply, for example, that the probability of having the first transmitter up and the last two transmitters down is 0.0313. The probability that all the transmitter units are up or $K = 3$ is 0.5168. In the next section, we complicate the situation by allowing different components within a system.

3.3.4 Case IV - M Radios With Different Components And Redundancy

To expand on Case III, assume that the transmitter units, made up of r_i transmitters each, where $r_i = r$ for convenience, are one component-type of a radio or device.

The second component-type for these radios is a receiver (see Figure 16 below).

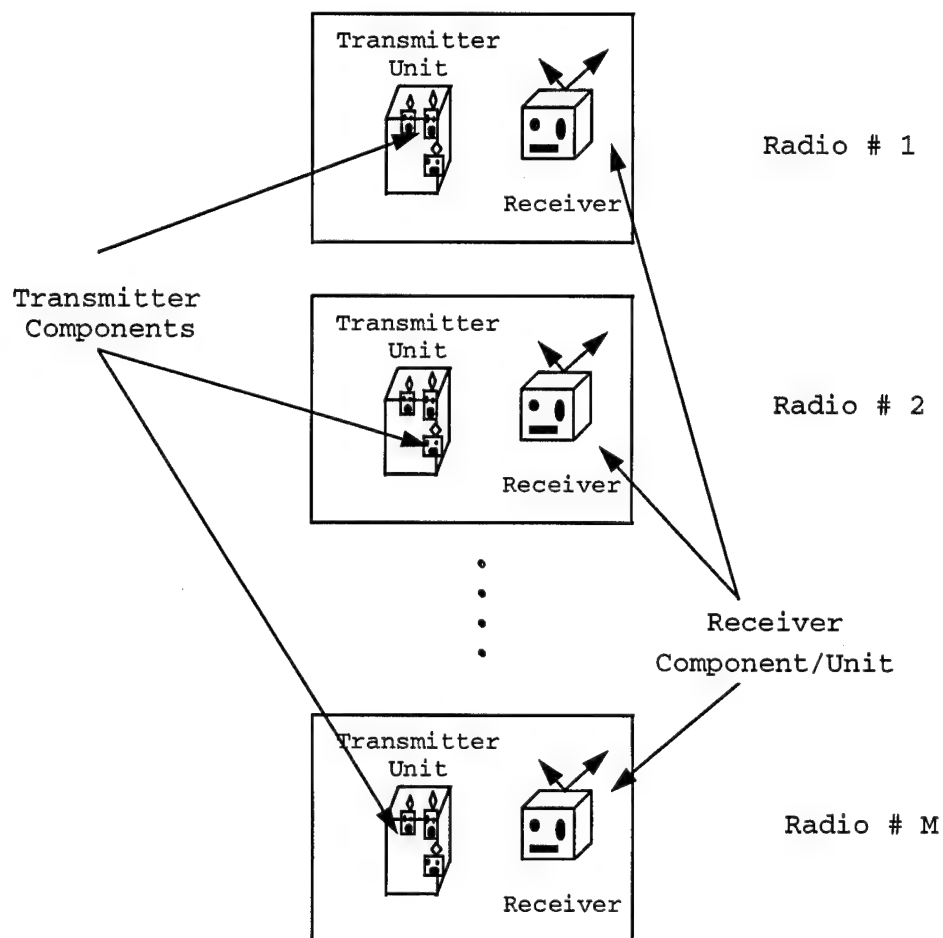


Figure 16. Case IV: M radio operating system

The transmitter unit and receiver together comprise a radio and there are M radios operating. A failure distribution is needed for each component-type in the radios. This is done by simply generalizing equation 3-14 for multiple

components. The failure distribution of component-type j for the first $1, 2, \dots, k$ units up and the $k+1, k+2, \dots, M$ units down is calculated using:

$$U_j(k) = \sum_{x=0}^{\infty} P_j\{X = x\} \sum_T \text{hyp}(y_1, y_2, \dots, y_M | x) \quad I \leq k \leq M \quad (3-15)$$

where

j - The component-type number

T - The set of part failure combinations that result in the first k units of type j up and the rest down

I - The minimum number of systems (Radios) that must be available

The results for each component-type are then combined into a system failure distribution. For this case study, there are two different component-types, a transmitter unit and a receiver, that make up a radio or device. The probability that all M radios or devices are up and none are down, $U(M)$, is given by:

$$U(M) = U_1(M) * U_2(M) \quad (3-16)$$

Below, $U(M-1)$ is the probability that the first $M-1$ radios are up and radio M is down.

$$U(M-1) = U_1(M-1) [U_2(M-1) + U_2(M)] + U_1(M) * U_2(M-1) \quad (3-17)$$

This equation results from the component-type combinations (transmitter and receiver) that create the first $M-1$ radios up and the M th radio down. Therefore, the first term in the equation above is created by the two cases where either both components have the first $M-1$ components up or the first component has the first $M-1$ up and the second component has all M up, both of which results in the first $M-1$ radios up and the M th radio is down. The equation for the first $M-2$ radios up and the $M-1$ and M th radio down gets more complex. It is given below:

$$\begin{aligned} U(M-2) = & U_1(M-2) [U_2(M-2) + 2U_2(M-1) + U_2(M)] \\ & + 2U_1(M-1) [U_2(M-2) + U_2(M-1)] \\ & + U_1(M) * U_2(M-2) \end{aligned} \quad (3-18)$$

The logic for building this equation, however, is the same as explained above. Simply, enumerate all combinations

that produce the first M-2 radios up and the M-1 and Mth radios down. For example, the first factor of two ($U_1(M-2) * 2U_2(M-1)$) is a result of the fact that with two radios down for transmitter units, there are two ways to select the radio that is also down for the receiver. This equation building process continues until the minimum number of acceptable operating radios or devices, I, is reached.

The following example builds on the numeric example from the previous section. There is one receiver on each radio. The receiver has a failure rate of 0.2 failures per hour and a repair rate of 1 repair per hour. This results in a value for f of 0.2. Assume there is one receiver spare. Using equation 3-12 results in the receiver backorder probability distribution below:

Table 12: Receiver Failure Probability Distribution

X	P(X)
0	0.5540
1	0.3324
2	0.0997
3	0.0133
4	0.0007

The probability of the first k receivers being up and the $k+1, k+2, \dots, M$ receivers being down is calculated using equation 3-15 and the results are presented below:

Table 13: Receiver Probability Distribution of First K Up
and $M-k$ Down

K	$U_2(K)$
0	0.0007
1	0.0044
2	0.0332
3	0.8864

Using equation 3-17, we can calculate the probability of the first two radios being up and the last radio being down, $U(2)$, as follows:

$$\begin{aligned}
 U(2) &= U_1(2) [U_2(2) + U_2(3)] + U_1(3) * U_2(2) \\
 &= 0.1341
 \end{aligned}$$

This gives a probability of the first two radios being up and the last radio being down of 0.1341. The results of these calculations can now be used to form radio system availabilities. This is discussed in the next section.

3.3.5 Calculating System Availabilities

Based on the previous calculations of the number of operating radio probabilities, an overall radio system availability can be computed. Since $U(k)$ is the probability that the first k radios are operating while the remaining $k+1$ to M are down, the availability, $A(k)$, of at least k radios operating can be found by simply computing the number of ways k up radios can be selected from the M radios. This results in the following equation:

$$A(k) = 100 [U(M) + \binom{M}{M-1} U(M-1) + \dots + \binom{M}{k} U(k)], \quad 1 \leq k \leq M$$

(3-19)

To continue using our example from the previous sections, assume we are interested in the availability of two or more radios in this system, A(2). The calculations are shown below:

$$A(2) = 100[U(3) + \binom{3}{2}U(2)]$$

$$A(2) = 100[0.4580 + 3 * 0.1341]$$

$$A(2) = 86.04\%$$

This implies that the expected availability of two or more radios for this system is 86.04% or the radio system at the current sparing levels can expect to have at least two radios operating about 6.88 hours of each 8 hour operating day.

This approach provides a better analytic solution than the Dyna-METRIC model for finite-calling population problems with item redundancy and will more accurately distribute component failures but it can be further refined by providing for the interaction between the failure and repair patterns of the different component-types. This is the topic of the next section.

3.3.6 Component Failure Feedback

The preceding approach handles the finite-population demand issue (i.e. failure rate decreases as failures increase) within a component-type but fails to consider the impact each component-type's failures have on the other failures in the radio. For example, if the radio transmitter unit is expected to break 3 of 4 radios, then the receiver will not accumulate the same number of operating hours or failures as it would if the radio transmitter unit did not exist in the radio. Therefore, some type of adjustment should be considered to account for this inter-component interaction. An adjustment of the other component-type's failure rates based on the highest failing component-type's expected availability will reflect this interaction. The expected number of operating units, $E[O_j]$, would be calculated for each component-type j using:

$$E[O_j] = \left[\sum_{k=0}^M \binom{M}{M-k} U_j(M-k) * (M-k) \right] \quad (3-20)$$

Then the component-type with the lowest number of expected operating units would be used to adjust the demand rates of the remaining component-type. This research used

the availability of this lowest component-type as the adjustment factor. This availability is then multiplied by the arrival rate of all the other component-types to obtain a new utilization rate. This equation is shown below:

$$f_{j \text{ new}} = f_{j \text{ old}} * E[O_L] / M \quad (3-21)$$

This new utilization rate, $f_{j \text{ new}}$, will be used to recompute all the other component-type's $U_j(k)$ and then the system availability recalculated.

Building on the example used in this chapter, the $E[O_j]$ for each component must be calculated. This is done below:

Transmitter Unit - Component #1

$$\begin{aligned} E[O_1] &= \left[\sum_{k=0}^3 \binom{3}{3-k} U_1(3-k) * (3-k) \right] \\ &= (0.5168) * (3) + (3) * (0.1272) * (2) \\ &\quad + (3) * (0.0313) + (0.0077) * (0) \\ &= 2.4075 \end{aligned}$$

Receiver - Component #2

$$\begin{aligned}
E[O_2] &= \left[\sum_{k=0}^3 \binom{3}{3-k} U_2 (3-k) * (3-k) \right] \\
&= (0.8864) * (3) + (3) * (0.0332) (*2) \\
&\quad + (3) * (0.0044) + (0.0007) * (0) \\
&= 2.8716
\end{aligned}$$

Based on these results, the transmitter unit is the pacing failure item and its availability should be used in equation (3-21) to adjust the other utilization rates. Therefore, the $f_{2 \text{ new}}$ is:

$$\begin{aligned}
f_{2 \text{ new}} &= f_{2 \text{ old}} * E[O_1] / M \\
&= 0.2 * (2.4075 / 3) \\
&= 0.1605
\end{aligned}$$

Then the system availability would be recomputed using the new utilization rates. This gives a new radio availability for two or more radios being up of 87.29%. If the number of components is more than two, then check to see if the increase in $A(M)$ ($A_{\text{current}}(M) - A_{\text{last}}(M)$) is more than 0.001. If it is, then select the second lowest $E[O_j]$ value, recompute the $f_{j \text{ new}}$ values, and compute new availabilities. When the change in $A(M)$ is less than 0.001, then stop the

process and report the availability results. The next section will summarize the method.

3.3.7 Summary Of Method And Preliminary Results

The following figure and step-by-step guide summarize the method covered in the previous sections.

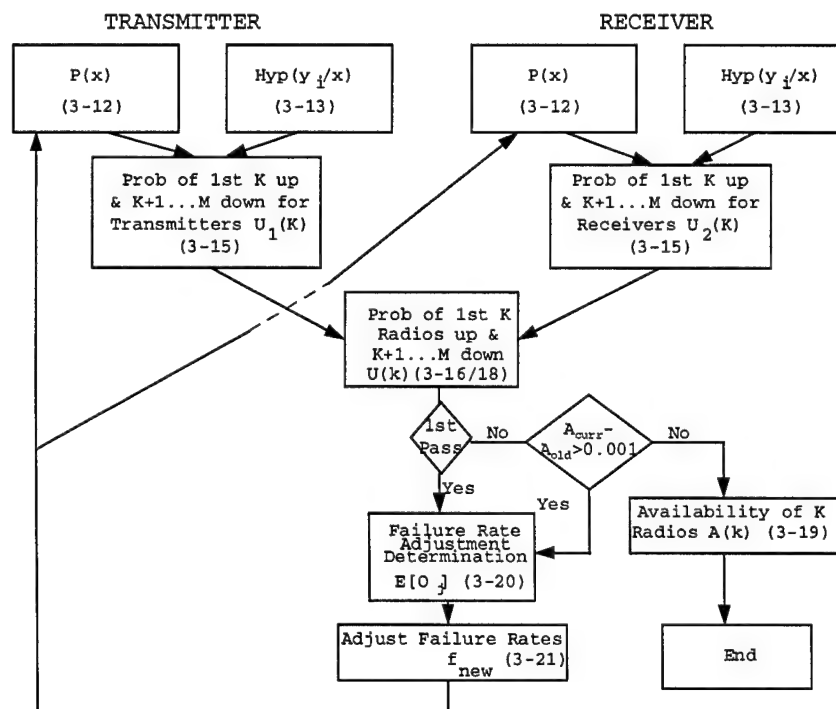


Figure 17. Flow of model calculations

Research Methodology

Step #1: Calculate the component failure distributions, $P(X)$, and the hypergeometric distribution of

failures $\text{Hyp}(y/x)$ for each component (eq. 3-12,13).

Step #2: Calculate the probability that the first k of each component are up and the $k+1, k+2, \dots, M$ are down, $U_j(k)$ (eq. 3-15).

Step #3: Calculate the probability that the first K pieces of the system are up and the $K+1, K+2, \dots, M$ pieces are down, $U(k)$ (eq. 3-17).

Step #4: If first pass or $A_{\text{current}}(M) - A_{\text{last}}(M) > 0.001$, go to step 5. Otherwise, go to step 8.

Step #5: Calculate the expected number of operating components of each component-type, $E[O_j]$ (eq. 3-20).

Step #6: Selecting the smallest $E[O_j]$ value in step 5, calculate the new utilization rates, $f_{j \text{ new}}$, for all the other component-types (eq. 3-21).

Step #7: Go to step 1 and repeat the steps for all components except the component whose $E[O_j]$ was used.

Step #8: Calculate the availability, $A(k)$, for the system based on the desired number of operating devices in the system (eq. 3-19).

Calculations are completed.

3.4 Computational Efficiency

Although the research model discussed in the previous section provides a good predictor of the modeled environment (see sec 4.2), when actually implemented using computer code, one problem becomes immediately obvious. When large numbers of parts are considered, the computer run times become significant. This problem occurs for as few as forty parts. The table below gives a summary of the run lengths in CPU hours for selected numbers of components. The computer run times are from a SPARCenter 2000 computer running Solaris 2.4. The times for component sizes over 30 are projected times based on the average time per computation for the runs under 30. These computation figures are based on eq. 3-17 but they represent the bulk of the computations necessary for this research method.

Table 14 : Growth in Computations and Run Lengths

Components	# Of Computations	CPU Hours
5	31.000000	7.368386E-08
10	1023.000000	2.431568E-06
15	32767.000000	7.788385E-05
20	1048575.000000	2.492357E-03
25	3.355443E+07	7.975550E-02
30	1.073742E+09	2.552176
35	3.435974E+10	81.669620
40	1.099512E+12	2613.428000
45	3.518437E+13	83629.700000
50	1.125900E+15	2676150.000000
55	3.602880E+16	8.563682E+07
60	1.152921E+18	2.740378E+09
75	3.777893E+22	8.979670E+13
80	1.208926E+24	2.873494E+15
85	3.868563E+25	9.195183E+16
90	1.237940E+27	2.942458E+18
95	3.961408E+28	9.415865E+19
100	1.267651E+30	3.013077E+21

This problem occurs because of the multiplicative effect that occurs as the number of parts grows in equation 3-16

through 3-18 which are used to compute the $U(k)$. The code used to generate these projections is given below. The subroutines called are the same as those called by the research model and can be found in appendix A:

```

PROGRAM MAIN
INTEGER X,Q,STKS,LRUAV,NOFAIL,R
REAL*8 BORDER(40,0:40),HYPERT1(1,3,0:5,0:20)
1,U(40,0:40),UTOT(0:20),HYPER(100,0:100)

OPEN(UNIT=7,FILE='TIME.INP',STATUS='OLD')
OPEN(UNIT=8,FILE='TIME.OUT',STATUS='OLD')
READ(7,*)RUNS
READ(7,*)ADJUST

9  R = 100
   CALL HYPERGEO(R,HYPER)
   PRINT *, 'THIS IS THE NO. OF CALCULATIONS & TIME FOR N'
   WRITE(8,*) 'THIS IS THE NO. OF CALCULATIONS & TIME FOR N'

   DO 10 I=1,RUNS
     READ(7,*)NUMBER

C    R - CREATES THE RANGE FOR HYPER MATRIX
     SUM=0
     TIME=0
     DO 17 A=1,R
       SUM=SUM+HYPER(NUMBER,A)
       TIME=TIME+HYPER(NUMBER,A)*ADJUST
17  CONTINUE
PRINT *,SUM,TIME,NUMBER
   WRITE(8,*) NUMBER,SUM,TIME
10  CONTINUE
     RETURN
   END

```

Figure 18: Computation code for number/run times

As the reader can see from the table above, as the number of components grows, the number of computations and run times grows rapidly! One simple solution that will reduce

run times is to change the algorithm slightly to avoid computing the $U(k)$ more than once. Since the feedback mechanism isn't dependent on the results of the $U(k)$ computations, simply compute the $E[O_j]$ earlier in the process and only compute the $U(k)$ once. The new algorithm would look like this:

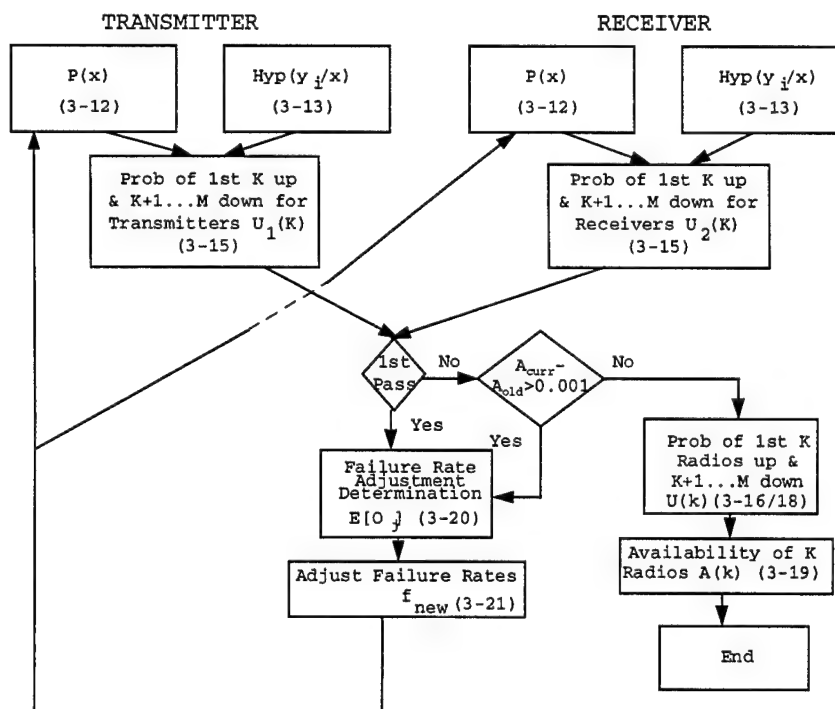


Figure 19. Flow of model calculations (revised)

Research Methodology (Revised)

- Step #1: Calculate the component failure distributions, $P(X)$, and the hypergeometric distribution of failures $\text{Hyp}(y/x)$ for each component (eq. 3-12,13).
- Step #2: Calculate the probability that the first k of each component are up and the $k+1, k+2, \dots, M$ are down, $U_j(k)$ (eq. 3-15).
- Step #3: If first pass or $A_{\text{current}}(M) - A_{\text{last}}(M) > 0.001$, go to step 4. Otherwise, go to step 7.
- Step #4: Calculate the expected number of operating components of each component-type, $E[O_j]$ (eq. 3-20).
- Step #5: Selecting the smallest $E[O_j]$ value in step 5, calculate the new utilization rates, $f_{j \text{ new}}$, for all the other component-types (eq. 3-21).
- Step #6: Go to step 1 and repeat the steps for all components except the component whose $E[O_j]$ was used.
- Step #7: Calculate the probability that the first K pieces of the system are up and the $K+1, K+2, \dots, M$ pieces are down, $U(k)$ (eq. 3-17).

Step #8: Calculate the availability, $A(k)$, for the system based on the desired number of operating devices in the system (eq. 3-19).

Calculations are completed.

This change cuts the number of computations almost in half. The next two sections discuss two additional approaches to reducing the run time problem.

3.4.1 Component Batching

Careful study of this problem reveals that each term in the computation of the $U(k)$'s for a large number of components is generally very small (i.e. represents only a small part of the tail of the distribution). One approach to reducing the run times is to ignore or cut away small portions of the tails of these distributions, thereby sacrificing some computational accuracy but significantly reducing run times. This was done by "batching" the parts into smaller groups for the $U(k)$ computations, performing the $U(k)$ calculations, and then taking the "batched" results and combining them into the system $U(k)$. The results in the following tables show that this technique

has only a very small impact on accuracy but a serious impact on run times.

The columns headed by "computed using all" and "computed using batches" describes how the values were computed for each table scenario and row. The table rows are built in sets of two. The first row represents the probability of all M aircraft being up and the second is the probability of M-1 aircraft being up. The first column also shows how the data was batched in the two-row set and below this, the value of M-1. The table shows run times and the differences and percentage error between the values compiled by batches and those computed with all the items in one group. All the runs were made on a Dell Low Profile Pentium ISA/PSI system with a Intel Comparative Microprocessor Performance (iComp) index rating of 735/90 and an internal clock speed of 90 Mhz except the 8/25 run in Table 20 which is in CPU minutes for the SPARCenter computer mentioned earlier. As an example, consider Table 16. The third and forth rows beginning 2/10 and 2 A/C represent the probabilities of 3 aircraft being up (row 3) and 2 aircraft being up (row 4). The batched results were computed using a 2/10 scheme. This means there were two batches of 10 components each. The difference in results is very small. The other results are similar. The only

exception to these results is that the 0 aircraft results, $A(0)$, in Tables 19 and 20 under the computed using all column were computed using $1 - A(1)$ as the result rather than the research method. This was based on the fact that the probability distribution must sum to one.

Table 15: Batch Results for 20 Components and 1 Aircraft

	Run time	Results for 20 Parts and 1 aircraft		
		Computed using all	Computed using batches	difference/% error
All	0:12	0.9543325980446936	0.9543325980446936	0 / 0
0 A/C		0.0456674019553064	0.04566740195529483	1.157 E-14 / 2.5 E-11
2/10	<1 sec	0.9543325980446936	0.9543325980446934	last digit
0 A/C		0.0456674019553064	0.04566740195530677	-3.7 E-16 / -8.1 E-13
4/5	<1 sec	0.9543325980446936	0.9543325980446931	last digit
0 A/C		0.0456674019553064	0.04566740195530664	-2.4 E-16 / -5.2 E-13

Table 16: Batch Results for 20 Components and 3 Aircraft

	Run time	Results for 20 Parts and 3 aircraft		
		Computed using all	Computed using batches	difference/% error
All	0:11	0.8691590890415416	0.8691590890415416	0 / 0
2 A/C		0.04159161864921457	0.04159161864921457	0/0
2/10	<1 sec	0.8691590890415416	0.8691590890415418	last digit
2 A/C		0.04159161864921457	0.04159161864919274	2.183 E-14 / 5.2 E-11
4/5	<1 sec	0.8691590890415416	0.8691590890415417	last digit
2 A/C		0.04159161864921457	0.04159161864919273	2.184 E-14 / 5.2 E-11

Table 17: Batch Results for 25 Components and 1 Aircraft

	Run time	Results for 25 Parts and 1 aircraft		
		Computed using all	Computed using batches	difference/% error
All	6:25	0.9251115113665423	0.9251115113665423	0 / 0
0 A/C		0.0748884886334577	0.07488848863287233	1.42537 E-12/1.9 E-9
3/10	<1 sec	0.9251115113665423	0.9251115113665421	last digit
0 A/C		0.0748884886334577	0.07488848863345815	-4.5 E-16 / -6.0 E-13
5/5	<1 sec	0.9251115113665423	0.9251115113665418	last 2 digit
0 A/C		0.0748884886334577	0.07488848863345800	-2.3 E-16 / -3.0 E-13

Table 18: Batch Results for 25 Components and 3 Aircraft

	Run time	Results for 25 Parts and 3 aircraft		
		Computed using all	Computed using batches	difference/% error
All	6:25	0.7917393952469072	0.7917393952469072	0 / 0
2 A/C		0.06409191321551984	0.06409191321551984	0/0
3/10	<1 sec	0.7917393952469072	0.7917393952469073	last digit
2 A/C		0.06409191321551984	0.06409191321598043	-4.6 E-12 / -7.1 E-10
5/5	<1 sec	0.7917393952469072	0.7917393952469072	0/0
2 A/C		0.06409191321551984	0.06409191321598043	-4.6 E-12 / -7.1 E-10

Table 19: Batch Results for 100 Components and 1 Aircraft

	Run time	Results for 100 Parts and 1 aircraft		
		Computed using all	Computed using batches	difference/% error
10/10	1 sec	0.2105289067195615	0.2105289067195615	0 / 0
0 A/C		0.7894710932804385	0.7894710932804411	2.6 E-15 / 3.3 E-13
5/20	2:35	0.2105289067195615	0.2105289067195616	last digit
0 A/C		0.7894710932804385	0.7894710932792852	1.2 E-12 / 1.5 E-10
4/25	1:15:00	0.2105289067195615	0.2105289067195615	0/0
0 A/C		0.7894710932804385	0.7894710932485831	3.186 E-11 / 4.0 E-9
20/5	30 sec	0.2105289067195615	0.2105289067195613	last digit
0 A/C		0.7894710932804385	0.7894710932787855	1.653 E-12 / 2.1 E-10

Table 20: Batch Results for 200 Components and 1 Aircraft

	Run time	Results for 200 Parts and 1 aircraft		
		Computed using all	Computed using batches	difference/% error
20/10	25 sec	0.0484630067792717	0.0484630067792717	0 / 0
0 A/C		0.9515369932207282	0.951536993220114	6.1 E-13/6.5 E-11
10/20	4:35	0.0484630067792717	0.048463006779271	0/0
0 A/C		0.9515369932207282	0.951536993221534	-8.1 E-12/-8.5 E-10
8/25	36*	0.0484630067792717	0.048463006779271	0/0
0 A/C		0.9515369932207282	0.951536993179098	3.2 E-11 /4.0 E-9
25/8	18:25	0.0484630067792716	0.048463006779271	0/0
0 A/C		0.9515369932207283	0.951536993156492	6.4 E-11 /6.8 E-09

*CPU minutes vs. other times are interactive times

The components in these runs were batched in a random order and therefore, accuracy loss could be further reduced by rank-ordering the components based on their respective $U_j(k)$ values before batching them, but this does not seem to be necessary due to the small accuracy loss. Now that we know that batching will reduce run times without significant losses in accuracy, what is the optimum batch size and the optimum number of batches in terms of run times? Since the total program run time is directly related to the computation of the $U(k)$'s, minimizing the number of cycles through the $U(k)$ computations will also minimize the program run time. To find the minimum number

of computations, the following non-linear integer programming (NLIP) problem can be formed:

$$\text{Min } Z = \sum_{m=1}^y \binom{Y}{m} + y \sum_{n=1}^x \binom{X}{n}$$

Subject to:

$$y \geq 0$$

$$x \geq 0$$

$$x * y \geq \text{LRU}$$

x , y , & LRU are integers

where

Z - The number of cycles through the computations

x - The batch size or number of components in each set of computations of $U(k)$

y - The number of batches of components of size X

LRU - The number of components in the data set

This problem could also be subject to other constraints such as:

$$x = \text{LRU for } \text{LRU} \leq 20$$

or other such restrictions on x and y values to eliminate trivial or needless batching.

One difficulty in solving any non-linear programming problem is finding a good set of starting conditions. It just so happens that for this problem, a simple set of starting conditions exists which has two advantageous properties. First, it can be shown that from this starting point one-half of the possible solution space can be ignored (see proof below) and second, this point is either the optimum solution or very close to the optimum such that the optimum can be found in only a few steps through a search algorithm. This starting point is $X = Y = \sqrt{LRU}$. The results in the table below present the optimum batch size, the optimum number of batches, the number of steps to find the optimum solution, and the number of computations for the $U(k)$'s for various component data set sizes that also minimize the research algorithm run times.

Table 21: Optimum Number of Batches and Batch Size

No. of Components	No. of Batches	Size of Batch	No. of Computations	Cycles to Optimum
25	5	5	1.86000000E+02	1
50	9	6	1.07800000E+03	2
75	11	7	3.44400000E+03	3
100	12	9	1.02270000E+04	2
125	13	10	2.14900000E+04	2
150	14	11	4.50410000E+04	2
175	15	12	9.41920000E+04	2
200	16	13	1.96591000E+05	2
225	18	13	4.09581000E+05	3
250	18	14	5.57037000E+05	3
275	19	15	1.14686000E+06	3
300	20	15	1.70391500E+06	3
325	21	16	3.47338600E+06	3
350	21	17	4.84964200E+06	3
375	21	18	7.60215400E+06	2
400	23	18	1.44178960E+07	3
425	23	19	2.04472080E+07	3
450	24	19	2.93601030E+07	3
475	24	20	4.19430150E+07	3
500	25	20	5.97688070E+07	3

Table 21 cont.

525	25	21	8.59832070E+07	3
550	25	22	1.38412007E+08	2
575	25	23	2.43269607E+08	2
600	27	23	3.60710117E+08	2
625	28	23	5.03316452E+08	3
650	28	24	7.38197476E+08	2
675	29	24	1.02341012E+09	3
700	28	25	1.20795955E+09	2
725	29	25	1.50994941E+09	3
750	30	25	2.08037478E+09	3
775	30	26	3.08700774E+09	3
800	31	26	4.22785830E+09	3
825	31	27	6.30823309E+09	3
850	32	27	8.58993434E+09	3
875	32	28	1.28849016E+10	3
900	33	28	1.74483046E+10	3
925	32	29	2.14748352E+10	2
950	33	29	2.63066736E+10	3
975	34	29	3.54334791E+10	3
1000	35	29	5.31502192E+10	4

The computer code used to solve this NLIP problem is in the figure below.

```

PROGRAM MAIN
  INTEGER Q,STKS,LRUAV,NOFAIL,R,LRU,COUNT
  REAL*8 BORDER(40,0:40),HYPERT1(1,3,0:5,0:20)
  1,U(40,0:40),UTOT(0:20),HYPER(100,0:100),SUMTOT(100)
  2,X,Y,MINX,MINY,OPS,SQRTS

  OPEN(UNIT=7,FILE='NLIP.INP',STATUS='OLD')
  OPEN(UNIT=8,FILE='NLIP.OUT',STATUS='OLD')
  READ(7,*)RUNS
  READ(7,*)LRU
  READ(7,*)ADJUST
  9  R = 100
  CALL HYPERGEO(R,HYPER)

  DO 10 I=1,RUNS
  READ(7,*)NUMBER
  C  R - CREATES THE RANGE FOR HYPER MATRIX
  SUM=0
  TIME=0
  DO 17 A=1,R
    SUM=SUM+HYPER(NUMBER,A)
    TIME=TIME+HYPER(NUMBER,A)*ADJUST
  17 CONTINUE
  SUMTOT(NUMBER)=SUM
  PRINT *,SUM,TIME,NUMBER
  WRITE(8,*) NUMBER,SUM,TIME

  10 CONTINUE
  SQRTS= SQRT(LRU)
  SQRTS= AINT(SQRTS)
  IF (SQRTS*SQRTS.GE.LRU) THEN
    X=SQRTS
    Y=SQRTS
    CALL OPTIM(X,Y,SUMTOT,MINX,MINY,OPS,LRU)
  ELSE IF (SQRTS*(SQRTS+1).GE.LRU) THEN
    X=SQRTS
    Y=SQRTS+1
    CALL OPTIM(X,Y,SUMTOT,MINX,MINY,OPS,LRU)
  ELSE
    X=SQRTS+1
    Y=SQRTS+1
    CALL OPTIM(X,Y,SUMTOT,MINX,MINY,OPS,LRU)
  ENDIF
  PRINT *, 'COMPONENTS  MIN Y  MIN X  OPERATIONS  COUNT'
  PRINT 100, LRU,MINY,MINX,OPS,COUNT
  WRITE(8,100) LRU,MINY,MINX,OPS,COUNT
  100 FORMAT(4X,I4,',',F3.0,',',F3.0,',',1P,E17.8,',',I2)
  RETURN
  END
C-----

```

```

SUBROUTINE OPTIM(X,Y,SUMTOT,MINX,MINY,OPS,LRU,COUNT)

REAL*8 X,Y,SUMTOT(100),MINX,MINY,OPS,CURRENT
INTEGER LRU,I,J,COUNT

CURRENT=SUMTOT(Y)+Y*SUMTOT(X)
MINX=X
MINY=Y
OPS=CURRENT
Y=Y+1
X=X-1
COUNT=0

DO 74 J=1,30
COUNT=COUNT+1
DO 50 I=1,30
IF (X*Y.GE.LRU) THEN
CURRENT=SUMTOT(Y)+Y*SUMTOT(X)
EXIT
ELSE
Y=Y+1
ENDIF
50 CONTINUE

IF (CURRENT.LT.OPS) THEN
OPS=CURRENT
MINX=X
MINY=Y
Y=Y+1
X=X-1
ELSE
EXIT
ENDIF

74 CONTINUE
RETURN
END

```

Figure 20: NLIP solution algorithm

The other subroutines called are the same as in the research model and are listed in appendix A.

The following proof shows why $X = Y = \sqrt{LRU}$ is a good starting point for the algorithm search. This is true because from this point, any increase in X even with a compensating decrease in Y will always result in an

increase in the value of Z . This eliminates the need to search one-half of the potential solution space.

Proof:

Assume $X = Y$ in our objective function, Z , above. If we add one to either X or Y , Z will always increase.

Therefore, if we add one to X , we should subtract one from Y . This implies that the new Z would be:

$$\frac{\sum_{n=1}^{x-1} \frac{(x-1)!}{n![(x-1)-n]!} + (x-1) \sum_{n=1}^{x+1} \frac{(x+1)!}{n![(x+1)-n]!}}{\sum_{n=1}^x \frac{x!}{n!(x-n)!} + (x) \sum_{n=1}^x \frac{x!}{n!(x-n)!}}$$

Combining terms gives:

$$\frac{\sum_{n=1}^{x-1} \frac{(x-1)!}{n![(x-1)-n]!} + (x-1) \sum_{n=1}^{x+1} \frac{(x+1)!}{n![(x+1)-n]!}}{(x+1) \sum_{n=1}^x \frac{x!}{n!(x-n)!}}$$

Add the zero term to each sum and then subtract its value from the sum:

$$\frac{\left[\sum_{n=0}^{x-1} \frac{(x-1)!}{n![(x-1)-n]!} - 1 \right] + (x-1) \left[\sum_{n=0}^{x+1} \frac{(x+1)!}{n![(x+1)-n]!} - 1 \right]}{(x+1) \left[\sum_{n=0}^x \frac{x!}{n!(x-n)!} - 1 \right]}$$

and restate:

$$\frac{\left[\sum_{n=0}^{x-1} \binom{x-1}{n} - 1 \right] + (x-1) \left[\sum_{n=0}^{x+1} \binom{x+1}{n} - 1 \right]}{(x+1) \left[\sum_{n=0}^x \binom{x}{n} - 1 \right]}$$

the combinations then reduce to:

$$\frac{[2^{x-1} - 1] + (x-1)[2^{x+1} - 1]}{(x+1)[2^x - 1]}$$

If we take the limit as X approaches infinity, then

$$\lim_{x \rightarrow \infty} \frac{[2^{x-1} - 1] + (x-1)[2^{x+1} - 1]}{(x+1)[2^x - 1]}$$

becomes:

$$\lim_{x \rightarrow \infty} \frac{[2^{x-1} - 1]}{(x+1)[2^x - 1]} + \frac{(x-1)[2^{x+1} - 1]}{(x+1)[2^x - 1]}$$

and the first term is:

$$\lim_{x \rightarrow \infty} \frac{[2^{x-1} - 1]}{(x+1)[2^x - 1]} = 0$$

leaving only the second term:

$$\lim_{x \rightarrow \infty} \frac{(x-1)}{(x+1)} * \lim_{x \rightarrow \infty} \frac{[2^{x+1} - 1]}{[2^x - 1]}$$

where:

$$\lim_{x \rightarrow \infty} \frac{(x-1)}{(x+1)} = 1$$

and

$$\lim_{x \rightarrow \infty} \frac{[2^{x+1} - 1]}{[2^x - 1]} = 2$$

thus,

$$\lim_{x \rightarrow \infty} \frac{(x - 1)}{(x + 1)} * \lim_{x \rightarrow \infty} \frac{[2^{x+1} - 1]}{[2^x - 1]} = 2$$

Therefore,

$$\lim_{x \rightarrow \infty} \frac{[2^{x-1} - 1]}{(x + 1)[2^x - 1]} + \frac{(x - 1)[2^{x+1} - 1]}{(x + 1)[2^x - 1]} = 2$$

This implies that in any case where X and Y begin as equals, if one is added to X and Y stays the same or decreases by one the objective function, Z, will increase. Thus, this half of the potential solution space can be ignored. Figure 21 is a sample plot of Z for the feasible solution space for 25 components (LRU = 25). The Y-axis is the total number of computations.

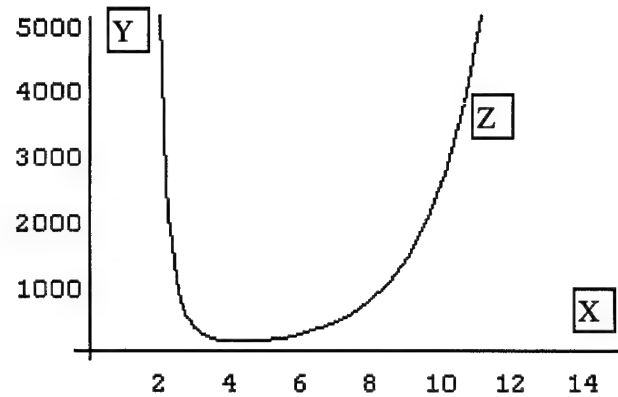


Figure 21: Plot of NLIP objective function

For this plot, the value of Y (no. of batches) is related to X by the constraint, $Y = 25/X$. This plot also shows how Z increases as X increases away from $X = Y = \sqrt{LRU}$.

Based on the results in this section, batching the components in an optimal way can significantly reduce run times and should be considered.

3.4.2 Accuracy Versus Speed Considerations

Further improvements in run times can also be achieved by reducing the number of components that are considered in the computations. This will have an obvious impact on the accuracy of the model and therefore, items dropped from the computations must be carefully selected. The method proposed to select these items is as follows:

- 1) Compute the $U_j(M)$ for each component-type (eq. 3-15)
- 2) Rank-order the components based on the results of #1
- 3) Determine an acceptable maximum accuracy loss, A_{\max}
- 4) Select the first item in the rank-ordered list
- 5) Compare the $U_j(M)$ of this item to the acceptable accuracy loss, A_{\max} , in step 3. If the $U_j(M)$ is greater than the accuracy loss, A_{\max} , then no component can be ignored in the computations and quit. Otherwise, set A_{tot} to $U_j(M)$ and go to step 6.
- 6) Select the next item in the rank-ordered list. Compute the total accuracy loss, A_{tot} by multiplying the $U_j(M)$ of this item to the previous A_{tot} .
- 7) Compare the A_{tot} to this point to the acceptable accuracy loss, A_{\max} , from step 3. If the A_{tot} is greater than the accuracy loss, A_{\max} , then this last component can not be ignored in the computations and achieve the desired accuracy. If $A_{\text{tot}} > A_{\max}$, the algorithm is completed and all items used to this point except the last can be ignored and still achieve desired accuracy loss. If the

acceptable loss, A_{\max} has not been reached, then go to step 6.

The A_{tot} computed above will be the maximum total accuracy loss. Stated another way, $A(M)$ will be increased by a factor of $1/A_{\text{tot}}$ if the items comprising A_{tot} are ignored. This is actually an upper bound on accuracy loss because these computations don't consider the interdependency of failures when computing the $U_j(M)$ and A_{tot} values. If it did, the $U_j(M)$ would very likely be smaller. Also, since the $\sum_{i=0}^M A(k) = 1$, any increase in $A(M)$ is a total reduction across all the other values of $A(k)$. It is, therefore, the maximum change among all the $A(k)$ because an increase in $A(M)$ will result in compensating reductions in all the other $A(k)$ values.

3.5 Research Model As Optimizing Tool

One capability that doesn't exist in any of the current models is the ability to quantify any additional capability that might be available if the availability goal is exceeded. For example, Dyna-METRIC will tell the user that he can fly his requested flying hours and maintain a

85% aircraft availability. But what if the availability goal is only 80%? Dyna-METRIC cannot tell you how many additional flying hours can be flown and yet still meet the availability goal. By adding a feedback mechanism to the research model and providing a user-input availability goal (Y), the research model can report the maximum operating hours per day per system, H, that can be achieved and still maintain the user availability goal (Y). This can be stated as:

$$\text{Maximize } H = \text{OHC} * \text{CPD}$$

$$\text{Subject to: } A(M-1) \geq Y \quad \text{for } M \geq 3$$

$$A(M) \geq Y \quad \text{for } M < 3$$

where:

H - Number of operating hour per system per day

Y - User-input availability goal

$A(M)/A(M-1)$ - are defined in eq. 3-16 & 17

OHC - Operating Hours per Cycle

CPD - Cycles Per Day

This is solved by adding the following step after step #8 to the research method described in section 3.4.

Step #9: If the availability of $M-1$ or more systems is greater than the availability goal, Y , (the availability of M systems is used for scenarios where $M < 3$), increment the daily operating hours per system by the user provided step size, G , and go to step #1. If the availability goal, Y , is not exceeded, then report the availability and operating hours for the last acceptable operating hour/availability combination. If no such combination exists, simply report this operating hour program and the expected system availabilities, $A(k)$'s.

Of course, this stopping rule could easily be changed to decrement operating hours if the availability goal could not be met by the current stock quantities to find the maximum operating hours while attempting to meet the availability goal.

3.6 Summary

This chapter presented the heart of the research. It showed key elements of previous work and then presented a

new method that overcomes many of the shortcomings of previous methods using finite-source queues and sampling without replacement. It also uses a feedback mechanism that attempts to account for the failure interdependency of components in a device. In addition, the chapter addresses one of the common problems with these approaches. This problem is the computational burden that often results from these solution methods. This chapter demonstrates several simple methods to reduce run times while limiting accuracy loss. It also presents a method for finding the maximum possible system operating hours while still maintaining a specified availability goal.

CHAPTER 4

RESULTS

4.1 Model Implementation

This section will discuss the implementation of the research method developed in Chapter Three and also some user issues that impact the ease of use of this method. This presentation will include a discussion of the model algorithm and a database management system used to create input files for this research. Although the case study used for validation involves aircraft, the use of this method is not restricted to this arena (see discussion Chapter 1).

4.1.1 The Research Algorithm

The following is a description of how the model developed in the previous chapter was actually implemented. The model has been written in FORTRAN and was compiled using Microsoft FORTRAN version 5.0 (1993). A complete listing of the program is in appendix A. The first set of operations performed by the program is to read-in a user-supplied input file that contains the system scenario and the component failure data. The figure below is a sample

input file used in section 4.2.4 for one of the B-52 test runs.

```

8      - Number of batches
5      - Batch Size
3      - Number of Aircraft
2      - Minimum Number of Up Aircraft
40     - Number of Components
0.50   - Number of Sorties per Day
14.4   - Number of Flying Hours per Sorties
0      - Cannibalization Flag
5895000395486 0.00670 1 1 1 0 0.00 2.00
1280001866298 0.00202 1 1 0 0 0.00 2.00
1660004098966 0.00023 1 1 1 0 0.00 2.00
6625004713174 0.00087 1 1 1 0 0.00 2.00
6610005300026 0.00065 2 2 1 0 0.00 2.00
6615005568510 0.00003 1 1 0 0 0.00 2.00
1560005926437 0.00020 1 1 0 0 0.00 2.00
1560006059700 0.00009 1 1 0 0 0.00 2.00
4320006847073 0.00013 2 2 1 0 0.00 2.00
6615007220964 0.00055 1 1 1 0 0.00 4.00
6340008117801 0.00031 3 3 1 0 0.00 2.00
4810008125687 0.00127 1 1 1 0 0.00 5.00
1560008976850 0.00058 1 1 0 0 0.00 2.00
1560008976853 0.00046 1 1 0 0 0.00 2.00
1560008976866 0.00020 1 1 0 0 0.00 2.00
1560008978397 0.00014 1 1 0 0 0.00 2.00
5895009049816 0.00101 1 1 1 0 0.00 2.00
6610010456372 0.00168 1 1 1 0 0.00 2.00
1560010598767 0.00006 1 1 0 0 0.00 2.00
5895010620002 0.00005 1 1 0 0 0.00 2.00
5895010701444 0.00026 1 1 0 0 0.00 2.00
5841010781344 0.00130 1 1 0 0 0.00 2.00
5865010887202 0.01070 1 1 1 0 0.00 2.00
1280011207217 0.00040 1 1 0 0 0.00 2.00
5895011327476 0.00077 2 2 0 0 0.00 2.00
6680011462360 0.00031 1 1 0 0 0.00 2.00
6680011469527 0.00035 1 1 1 0 0.00 2.00
1280011513174 0.00023 1 1 0 0 0.00 2.00
5895011525214 0.00072 1 1 0 0 0.00 2.00
1660011591246 0.00020 1 1 1 0 0.00 2.00
6615011655941 0.00254 1 1 1 0 0.00 2.00
1660011680330 0.00035 1 1 0 0 0.00 2.00
6220011987217 0.00040 1 1 1 0 0.00 2.00
1280012088417 0.00030 2 2 0 0 0.00 2.00
6605012094229 0.00489 2 2 3 0 0.00 2.00
6615012253746 0.00133 1 1 1 0 0.00 2.00
1280012270719 0.00081 1 1 1 0 0.00 2.00
1280012283930 0.00459 1 1 1 0 0.00 6.00
1280012285349 0.00008 1 1 0 0 0.00 5.00
5821012287058 0.00237 1 1 1 0 0.00 2.00

```

Figure 22: B52 research model input file

The first eight fields are defined on the figure. The remaining rows contain component data. Each row contains information for one component. The following is a description of each column or entry.

- 1) The component stock number
- 2) The failure rate for the component in hours
- 3) The quantity per unit
- 4) The minimum quantity per unit which allows the unit to still operate properly
- 5) The number of spare parts
- 6) The number of subcomponents in this component
- 7) The percentage of parts not repaired at this location
- 8) The average number of days to repair the component

The model then builds and stores a table of binomial coefficients for use throughout the rest of the program. Next, the model builds the Hypergeometric combinations (eq. 3-13). From Chapter Three, we're reminded that this can be done without reference to the failure or repair information of the components and therefore, can be computed once and stored. Because of the difficulty in writing this portion

of the algorithm, the maximum QPA and minimum QPA were limited. For most of the components being considered this was not an issue and won't effect the results of this research, although expanding this capability should be considered for future work.

At this point, the following series of calculations is carried out for each component-type. First, the failure distribution for each component-type is computed (eq. 3-12). Then the $U_j(k)$ for each possible k value is calculated using eq. 3-15. These values are stored and used to calculate the $E(O_j)$ for each component-type (eq. 3-20). This $E(O_j)$ is compared to all the other component $E(O_j)$ values and if it is the lowest $E(O_j)$, it is stored for later use. This process continues until all the $U_j(k)$'s for each component-type have been determined. The model also calculates the probability of all of the systems being up, $U(M)$, based on eq. 3-16.

Based on the component with the lowest $E(O_j)$, the process returns to recompute the $P(X)$'s and $U_j(k)$'s for each component-type based on their new f_{new} (see eq. 3-21). At the completion of this second pass through the computations, the value of the first $U(k)$ is compared to the new $U(k)$. If the difference is larger than the desired

accuracy limit (now set at 0.001), then the new second lowest $E(O_j)$ is stored and used to recompute the $U_j(k)$'s again. This process continues until the change in $U(M)$'s from one run to the next is smaller than the desired accuracy limit. When the difference is within the desired accuracy limit, the algorithm moves to the next phase of the computations.

The components are split into batches based on the input batch sizes and the system $U(k)$'s are calculated for each batch (eq. 3-16, 3-17). The results from each of these batches is stored and after all the batches have been computed, combined to create a set of system availabilities (eq. 3-19). A listing of the top five problem components is also given based on the component $E(O_j)$ values. A sample of the research model output is shown in the figure below. This output is for the input file shown earlier.

```

THIS IS THE NUMBER OF AIRCRAFT  3
THIS IS THE NUMBER OF LRUs ANALYZED  40
THIS IS THE NUMBER OF SRUs ANALYZED  0

THIS IS THE PROBABILITY OF ALL SYSTEMS UP -  54.6196649304E-02
THIS IS THE PROBABILITY OF  2 SYSTEMS UP -  11.8373048194E-02

This run had  17 cycles through the components.

The availability of  3 systems is  54.6196649304E+00
The availability of  2 or more systems is  90.1315793888E+00
The availability of exactly 2 systems is  35.5119144583E+00

This is a list of the top 5 problem components
  Stock Number      E[O]
1280012283930      2.874E+00
1280001866298      2.919E+00
5865010887202      2.927E+00
5895011327476      2.940E+00
5841010781344      2.951E+00

```

Figure 23: B52 research model sample output

The preceding was a brief description of the how the research method described in Chapter Three was implemented. In the next section, other implementation issues are discussed.

4.1.2 Model User Issues

The following is a brief discussion of other model implementation concerns. One major issue to consider in using any of the models in this area is managing the large volumes of data and the general user-unfriendliness of that data. The figure below is a small sample of a data file provided by the Air Force for use in their Dyna-METRIC models. This file only contains the scenario and failure

information for ten components. This sample came from the Dyna-METRIC file from which the previous B52 research model input file was built.

[illegible]


```

1280001596180  WR  1 1001001000000098000009800200 0100 000000200 0100 0000
1280001596180  0800 0100 000000400 9999 00010515005150 00053353 77JF0  0
1280001596185  WR  1 10010010000000306000030600200 0100 000000200 0100 0000
1280001596185  0800 0100 000000500 9999 00040767007670 00065374 77JE0  0
1280001596188  WR  1 10010010000000384000038400500 0100 000000500 0100 0000
1280001596188  1100 0100 000001400 9999 00000665006650 00962599 77EA0  0
1280001866244  WR  1 1001001000000043000004300200 0100 000000200 0100 0000
1280001866244  0800 0100 000000800 9999 00000490004900 00012058 77DAV  0
1280001866298  WR  1 10010010000000202000020200200 0100 000000200 0100 0000
1280001866298  0800 0100 000000400 9999 00010626006260 00069634 77DCA  0
APPL
1095004752436  BARK00100
1095004752437  BARK00100
1095004882075  BARK00100
1095011003892  BARK00100
1240002362373  BARK00100
1280001596180  BARK00100
1280001596185  BARK00100
1280001596188  BARK00100
1280001866244  BARK00100
1280001866298  BARK00100
VTM
1095004752436  0 0100 0100 0100 0100 00001 000000 000000
1095004752437  0 0100 0100 0100 0100 00001 000000 000000
1095004882075  0 0100 0100 0100 0100 00001 000000 000000
1095011003892  0 0100 0100 0000 0100 00001 000000 000000
1240002362373  0 0100 0100 0100 0100 00001 000000 000000
1280001596180  0 0100 0100 0100 0100 00001 000000 000000
1280001596185  0 0100 0100 0100 0100 00001 000000 000000
1280001596188  0 0100 0100 0100 0100 00001 000000 000000
1280001866244  0 0100 0100 0000 0100 00001 000000 000000
1280001866298  0 0100 0100 0100 0100 00001 000000 000000
INDT
STK
1095004752436  1BARK00100 0001
1095004752437  1BARK00200 0001
1095004882075  1BARK00010 0001
1095011003892  1BARK00029 0001
1240002362373  1BARK00004 0001
1280001596180  1BARK00005 0001
1280001596185  1BARK00010 0001
1280001596188  1BARK00008 0001
1280001866244  1BARK00004 0001
1280001866298  1BARK00007 0001
INDC
1095004752436  75EAG  RELEASE, BONOP01002
1095004752437  75EAG  RELEASE, BONOP01002
1095004882075  75ACB  SELECTOR AAAA01002
1095011003892  75KAS  RACK, BOMB AAA01002
1240002362373  11RAB  CELL, OPTICAAA01002
1280001596180  77JF0  GENERATOR, AAA01002
1280001596185  77JE0  SERVO CONTAAA01002
1280001596188  77EA0  SCANNER ASAAA01002
1280001866244  77DAV  PANEL, INDIAAA01002
1280001866298  77DCA  ELECTRONICAAA01002
END

```

Figure 24: B52 Dyna-METRIC sample input file

The format for this file can be found in the version 4.6 supplement to Isaacson et.al. (1988). A sample of this format guide is provided in appendix B. As one can quickly

see, this file is not going to be easy to read or modify in this format. Therefore, a database manipulation system was developed using DBASE III+ (1991). The code used to build this system is in appendix D. The opening screen to this program is shown below.

```
+-----+
|               D I S S E R T A T I O N   D A T A   A I D               |
+-----+
|
|               1. EDIT KIT NAMES
|               2. UTILITIES
|               3. REVIEW COMPONENT DATA
|               4. COMPUTE MTBDs
|               5. COMPUTE DEMAND RATES
|               6. CHECK FOR DATA ERRORS (INC. INDENTURE RELATIONSHIPS)
|               7. BUILD MODEL FILE
|               8. REPORTS
|
|               0. EXIT
|
+-----+----- select 0 -----+
```

Figure 25: Opening screen to Dissertation Data Aid

This program allows the user to maintain up to 10 different aircraft units in one database (option one). The program also provides the capability to compute new failure rates based on the total number of failures and the total number of component operating hours (option four and five). Option six allows the user to check for common data errors such as unreasonable failure and repair rates. Option eight allows the user to build different information reports on the aircraft data. Options two, three, and

seven are the most often used options, therefore they will be discussed in more detail below.

The Utilities option (option two) has an opening screen that is shown in the figure below:

```
+-----+
|               U T I L I T I E S               |
+-----+
|
| 1. CONVERT DMAS FILE TO DATABASE FILE
| 2. RESET ALL APPLICATION FRACTIONS TO 1.00
| 3. RESET ALL REVIEWED FLAGS
| 4. PACK DATABASE (ELIMINATE DELETED ITEMS)
| 5. CHANGE PRINTER AVAILABILITY
| 6. READ DMAS COMPUTED QTYs INTO STK COLUMN
|
| 0. EXIT
|
+-----+ select 0 -----+
```

Figure 26: Utilities option opening screen

From this screen the user can read a Dyna-METRIC input file into the database (option one), reset or clear fields throughout the database (option two-five), and finally, the last option allows the user to read stock quantities computed within DMAS into the database (option six).

Returning to the main screen, option three allows the user to view and modify the individual elements of the component data. Below is a figure that contains a sample of this data manipulation screen.

```

NSN:5865010887202      WUC:76TB0 NOUN:RECEIVER-T      COST: 149983
COMP:ADJ STOCK: 20 QPA: 1 RCT: 2.00 NRTS: 0.00 DEPOT:SC DMD RATE: 0.01070
                      Min QPA: 1                      DEMANDS: 15
PERCENT APPLICATION:                      WADJ: 1.00      MTBD: 93.458
1111: 1.00                      FH CODE: A

```

```
LRU/SRU: L   MAINT (0-RR,1-RRR): 0
```

```
IF PART IS SRU, INDENTURES ARE SHOWN ON FOLLOWING SCREEN
```

```
CURRENTLY DELETED: N   REVIEW: Y
```

```
REMARKS: This is a new item.
```

Figure 27: Data manipulation screen

Finally, option seven from the main screen allows the user to create a number of different model input files from the information stored in the database. The selection screen for this option is shown below.

```

+-----+
|               V E R S I O N   S E L E C T O R               |
+-----+
|                                                                |
|               1. Dyna-METRIC VER 4.6                          |
|               2. Dyna-METRIC VER 6.0                          |
|               3. RESEARCH MODEL                               |
|                                                                |
|               0. EXIT                                          |
|                                                                |
+-----+----- select 0 -----+

```

Figure 28: Model selection output screen

Three different model input files can be built. They include a Dyna-METRIC version 4.6 file, a Dyna-METRIC

version 6.0 file, and an input file for the research model. The research model input file can also be easily modified for input into the simulation model discussed later by simply changing aspects of the first eight lines.

A typical use of this data management system might include the following steps:

- 1) Read in a Dyna-METRIC input file
- 2) Add computed or current on-hand stock to the database
- 3) Modify the demand rates and any other necessary component information
- 4) Build an input file for the research model
- 5) Run the research model using this input file

Using the tools developed in the last two sections, verification and validation of the research methodology can begin.

4.2 Verification and Validation

4.2.1 Verification And Validation Results

Now that a technique has been developed, it must be verified and validated. Verification was a continuous process that involved a number of steps. Each module, as it was developed, was tested to ensure appropriate input-output transformations were occurring. Most modules were tested with various sample input parameters to test general operating conditions. The hypergeometric calculations for determining the set T, however, were 100% tested because of the difficulty in implementing this logic. Finally, several full-scale test cases were run at the end of verification for both the component redundancy and no component redundancy cases to ensure model coding accuracy. Also, Mr. Ed Yellig, another Ph.D. candidate, compared the developed FORTRAN code to the model flow charts to ensure accurate model translation. HQ ACC, the research sponsors, have also looked at the model development and structure and agree that the research methodology models the target environment and that the approach is sound (Hale, 1995). The sponsor's only concern is with model run times which this research has effectively dealt with (see section 4.3).

Model validation, defined as raising the decision-maker's confidence to an acceptable level, will include a number of steps (Shannon, 1975). Checking the face-value of the model, the model structure and model assumptions is one step of the validation process. The other step involves checking input-output transformations with the current model and checking the predictive capability of the model against a real aircraft deployment. The structure and assumptions testing will be an iterative process involving the field experts. The field experts include members of Air Combat Command Headquarters. As developments are made, they will be checked with the agency mentioned above for credibility and accuracy. When this group is satisfied that the method is structurally sound, this portion of the validation will be complete. Once the method has been developed and the structure validated, it can be compared with reality. The data analysis section deals with this part of the research.

The potential users will provide the expert input for face-value and output validity tests. Measuring a decision-maker's confidence in a tool or method is a difficult issue. For any tool, confidence will vary by each individual based on their understanding of the tool, their willingness to take risks, and the importance of the

decision. It will also be affected by the person's stress level and the time constraints under which the person is working. For purposes of this research, the sponsors at HQ ACC/LGSW and the members of the research committee must be satisfied that the model does a competent job of modeling reality. Certainly, if the Air Force allows unit commanders to use this method for unit-level combat assessments in reporting to the Pentagon and Congress about military readiness, this will imply they have placed a high level of trust in the results it provides. This, of course, is a long term goal and not an immediate research requirement for measuring success because of the long period of time necessary for user confidence-building and the large user community that must be satisfied before such an implementation can take place.

Using data from normal peacetime deployments was considered as a source of data, but the problems in obtaining, controlling, and verifying this data were difficult. It was also likely that most of the restrictions and limitations observed in wartime deployments would not be observed in these peacetime deployments. Any or all of these issues would have made any data collected from these deployments of questionable value.

In order to build the test database, a simulation model was developed to create simulated "live" data. The simulation model in Lewis (1987) is used as a baseline for creating this simulated data. The problem with this model is it can only handle 40 components at a time. Aircraft in ACC are basically divided into two groups, fighters and "heavies". Fighters include aircraft such as the F-15, F-16, A-10 and F-111. Heavies include aircraft such as the B-52, B-1, E-3, and B-2. The following test parameters were established through work with HQ ACC and are documented in Miyares (1995) and its attachments. Two aircraft from each group based on the recommendations of HQ ACC have been selected. These aircraft are the B-52, E-3B, F-15, and F-16.

For each of these aircraft types, three aircraft sizes have been selected based on the user's requirements. Based on ACC advice, stock positions were set at three different positions: zero stock, fully authorized level, and the average fill percentage level based on the 9 March 1995 fill rates from the WSMIS/SAM report. Flight profiles were also provided by HQ ACC. The table below presents the flight profiles used in this study and the CSMS stock level fill percentages.

Table 22: Aircraft Flight Profiles and Stock Level Fill
Percentages

Aircraft Type	Squadron	Sortie Rate	Sortie Duration	Stock Fill Percentage
F15C	58th Ftr Sq	1.00	1.98	82%
F16C	68th Ftr Sq	1.0	1.98	64%
B-52H	20th Bomb Sq	0.5	14.4	89%
E-3B	30th Sq	0.5	13.4	78%

HQ ACC, as a research sponsor, has provided a great deal of input into the validation and testing process. This is an advantage because it improves the chances of final acceptance by the users. The next section discusses the analysis of the research results.

The inference of interest is the comparison of the research model to the simulation results. This comparison was done on an individual scenario and an overall model level. The comparisons were broken into two cases. The first has no component redundancy and the second allows component redundancy.

The comparisons were done as follows. For the individual scenario comparisons, a 95% confidence interval

was built for the simulation results. The research model result was then compared to the simulation result to see if it fell inside the confidence interval. This process will demonstrate whether certain scenarios or component data sets can impact or bias the accuracy of the model.

A two-sided hypothesis test was used to perform an overall assessment of the method performance. The following is a more detailed description of this test procedure. Samples from each model were collected. Each pair of runs was based on a common scenario. The availability result for the simulation model was subtracted from the research model availability result. This provided 60 differences for the aircraft availability measure. A two-sided hypothesis test was built to test for differences between the model outputs. The null hypothesis will be that there is no difference between the data sets while the alternative hypothesis will be that there is a statistically significant difference in the output data.

As a result of these tests, a discussion of similarities and differences between the data sets can be made. Also, conclusions and recommendations for future work and possible implementation of this method in the Air Force can be made.

Mace (1964) in his early work on sample size determination states that there are five important elements to a written statement of the experimental objectives. These include precise definitions of the following:

- 1) the process involved in the decision
- 2) the population(s) involved in the decision
- 3) the population parameter of interest to the decision-maker
- 4) the inference about the population parameter necessary for a decision
- 5) the precision in the experiment. This should be a value which is of practical significance.

The process involved in the decision-making process is described earlier in this document and involves predicting the impact of spares support on the deployment performance of a small unit of aircraft. The populations that were sampled are the outputs from the simulation model and the research model. Each of these samples use the same input data.

The following is a statement of the Central Limit Theorem (CLT) (Mendenhall, Scheaffer, and Wackerly, 1986):

Let Y_1, Y_2, \dots, Y_n be independent and identically distributed random variables with $E(Y_i) = \mu$ and $V(Y_i) = \sigma^2 < \infty$. Define

$$U_n = \sqrt{n} \left(\frac{\bar{Y} - \mu}{\sigma} \right) \quad \text{where} \quad \bar{Y} = \frac{1}{n} \sum_{i=1}^n Y_i \quad (4-1)$$

Then the distribution function of U_n converges to a standard normal distribution function as $n \rightarrow \infty$.

Mendenhall, Scheaffer, and Wackerly also present a proof of this theorem. Given that n , the number of test samples, will never reach infinity, how big must n be to get reasonable results? Mendenhall, Scheaffer, and Wackerly (1986) state that any n greater than 30 is usually sufficient to ensure U_n can be reasonably approximated by the normal distribution. Hines and Montgomery (1980) provide additional guidance by saying the Y_i generally fall into one of three categories. They are:

1. Well-behaved - The distribution of the Y_i have a bell-shaped density that is fairly symmetric. The n should be $n \geq 4$.
2. Fairly well-behaved - The distribution of Y_i has no prominent mode and looks somewhat like a uniform distribution. They suggest an $n \geq 12$.

3. Ill-behaved - The distribution of Y_i have most of their weight in the tails. They suggest an $n \geq 100$ as reasonable.

Based on this advice a sample size of 60 should satisfy the Central Limit Theorem.

The following two-sided test was taken from Hines and Montgomery (1980). It is a test of hypothesis for the mean of a Normal distribution where the variance of the distribution is unknown. This test assumes that the Y_i are normally distributed but Montgomery (1991) states that the test is not terribly sensitive to slight departures from the normality of the Y_i if the sample size is large (see discussion above on the CLT).

In order to test the hypothesis that the sample is from a normal distribution, a Shapiro-Wilks test was used. This test is shown below and was taken from Conover (1980):

Data: A random sample of size n , Y_1, Y_2, \dots, Y_n , taken from a population with some unknown distribution function, $F(y)$.

Assumptions: The sample consists of a random sample.

Hypotheses:

H_0 : $F(y)$ is a normal distribution with no specified mean and variance.

H_1 : $F(y)$ is not a normal distribution.

Test Statistic:

- 1) Compute the denominator, L , of the test statistic

$$L = \sum_{j=1}^n (y_j - \bar{Y})^2 \quad (4-2)$$

where \bar{Y} is the sample mean.

- 2) Order the sample from smallest to largest, $Y^{(1)} \leq Y^{(2)} \leq \dots \leq Y^{(n)}$.
- 3) Using the coefficients for this test, b_1, b_2, \dots, b_m , where m is equal to approximately $n/2$, the test statistic is:

$$W = \frac{1}{L} \left[\sum_{j=1}^m b_j (Y^{(n-j+1)} - Y^{(j)}) \right]^2 \quad (4-3)$$

Decision Rule: If W is less than the α value for the test's level of significance given in a table of Shapiro-Wilk test statistics such as in Conover (1980) Table A18, reject H_0 . This test statistic can be converted to an approximately normal random variable to be compared with

the normal distribution for a more accurate critical level. For this method, see Conover (1980). Now more about the two-sided hypothesis test from Hines and Montgomery (1980).

Let $D_i = (\text{research output}_i - \text{simulation output}_i)$ and \bar{D} and S^2 be the sample mean and variance, then the two-sided test is as follows:

$$H_0: \bar{D} = 0$$

$$H_a: \bar{D} \neq 0 \quad (4-4)$$

with the following test statistic:

$$t_o = \frac{\bar{D} - 0}{S/\sqrt{n}} \quad (4-5)$$

This test statistic follows a t distribution with $n-1$ degrees of freedom if H_0 is true. After t_o is calculated, H_0 is rejected if either

$$t_o > t_{\alpha/2, n-1} \quad (4-6)$$

or

$$t_o < - t_{\alpha/2, n-1} \quad (4-7)$$

where $t_{\alpha/2, n-1}$ and $- t_{\alpha/2, n-1}$ are the upper and lower $\alpha/2$ percentage points of the t distribution with $n-1$ degrees of freedom. This same two-sided test method can be shown to be equivalent to using the Likelihood Ratio test

demonstrated in Mendenhall, Scheaffer, and Wackerly (1986) which is a more general method.

To answer the question: "What constitutes a significant difference in a data set?" The military has defined a significant difference as ± 1.25 sorties per day and ± 1.0 aircraft for aircraft availability in Miyares (1995).

A significance level or type one error of $\alpha = 0.05$ is being used. This is the probability of rejecting H_0 when H_0 is true (Mendenhall, Scheaffer, and Wackerly, 1986). This value was also coordinated with the military as an acceptable value (Lewis, 1987) and has been revalidated by Miyares (1995). The type II error or β error is the probability of accepting H_0 when H_a is true. It is inversely related to the type I error for a fixed sample size (Mendenhall, Scheaffer, and Wackerly, 1986). Hines and Montgomery (1980) provide a method to determine the type II error for this hypothesis test method. When H_0 is false, the true value of the mean is $\mu + \delta$ and therefore the test statistic becomes:

$$t_o = \frac{\bar{D} - 0}{s/\sqrt{n}}$$

$$t_o = \frac{\frac{[\bar{D} - (0 + \delta)]\sqrt{n}}{\sigma} + \frac{\delta\sqrt{n}}{\sigma}}{S/\sigma}$$

$$t_o = \frac{Y + \frac{\delta\sqrt{n}}{\sigma}}{U} \quad (4-8)$$

Y and U are independent random variables and distributed $N(0,1)$ and $\sqrt{\chi^2_{n-1}/n - 1}$, respectively. But, because the second term in the numerator, $\delta\sqrt{n}/\sigma$, is non-zero, the numerator is now a $N(\delta\sqrt{n}/\sigma, 1)$ random variable. The distribution becomes a noncentral t distribution with $n-1$ degrees of freedom and a noncentrality parameter of $\delta\sqrt{n}/\sigma$. Therefore, $\beta = P\{-t_{\alpha/2, n-1} \leq t'_o \leq t_{\alpha/2, n-1}\}$ where t'_o denotes the noncentral t random variable. Operating characteristic (OC) curves plot β versus a parameter c defined as $c = |\delta|/\sigma$ for various sample sizes. Since the sample size is known, if a c value could be determined, then the type II error or β could be found. Since c depends on the unknown variance σ^2 , determining an initial β value will be difficult. Once the data is collected the sample variance S^2 can be used to estimate σ^2 and the β value calculated.

An initial value for β could also be calculated if the relative difference of $|\delta|/\sigma$ could be estimated. For example, if the experimenter is sensitive to small differences in the means, one could use a c value of less than one. If $c = 0.5$ for example, this would yield a β of 0.05 or the power of the test is 0.95, where the power of the test is defined as the probability of rejecting H_0 when H_0 is false (Power = $1 - \beta$). This result is based on the OC curve found in Appendix Chart VIe from Hines and Montgomery (1980).

4.2.2 Research Model Analysis

The following is an outline of the process used to create the research model results for comparison with the simulation model results.

As stated earlier, the Air Force uses two primary measures of merit to assess unit performance (Tripp et al., 1991). These measures include the aircraft availability and the percentage of actual sorties flown versus sorties tasked and both are based on average values. Since these are the Air Force standard measures, this study will also use them as the basis of comparison. Also, since the number of sorties flown is currently derived from the

average aircraft availability, the average availability will be used as the primary measure for validation.

In order to generate the appropriate test databases, a number of steps had to be taken. First, the Dyna-METRIC files had to be read into the database manager mentioned earlier. The table below summarizes the number of components in each database.

Table 23: Sizes of Test Databases

Aircraft Type	Number of Components
B-52H	347
E-3B	257
F-15C	139
F-16C	173

Because the simulation model is limited to analyzing only 40 parts at any time, a program to randomly select the parts from each aircraft database was used. It is shown in the figure below:

```
$DEBUG
      PROGRAM MAIN
      INTEGER X
      REAL SELECT, CHOSE
      OPEN(UNIT=7, FILE='PARTS', STATUS='OLD')
      X=165
      CALL SEED(X)
      DO 10 I=1,50
      CALL RANDOM(SELECT)
      CHOSE=AINT(X*SELECT)
      WRITE(7,*) CHOSE
10    CONTINUE
      RETURN
      END
```

Figure 29: Code to randomly select test components

Based on the output from this program, forty unique components from each aircraft-type were selected to be used in the model runs and analysis. One exception to this policy was made for the F-16's but this will be discussed later. A listing of the selected stock numbers, work unit codes, demand rates, repair cycle times and DMAS computed quantities (discussed later also) for each aircraft type are listed in Appendix B. After the components were selected, the component data was checked for reasonableness to include such things as demand rates greater than zero and less than one per sortie and repair times between two and six days. When this selection process was complete,

new DMAS input files were generated to compute standard Air Force authorized stock quantities. This must be done for two reasons. First, the Air Force does not maintain a listing of authorized stock levels for all the aircraft sizes this study is using and secondly, when the number of component-types being analyzed changes the authorized quantities also change. Since this research is only looking at forty components on each aircraft-type, the authorized quantities will be affected.

The new DMAS input files were input into DMAS. The same scenario that was used for the research and simulation runs was also used for the DMAS model runs. The figure below is a sample from the scenario page of DMAS for the 5 aircraft B-52 DMAS model run.

DMAS Version 3.3							
Deployment Computation Output							
Database: B52H				Section: AAA			

SCENARIO SUMMARY FOR TACTICAL AIRCRAFT							

Data Section: AAA				Date: 07-04-95		Time: 12:51	
NFMC #1 Days: 1-90				Goal: 1.0	Percentage: 20.00%		
DAYS	ACFT	SORTIES RATE	TOTAL	SORTIE DURATION	FLYING HOURS	MAX SORTIE RATE	ATTRITION RATE
1-90	5.00	0.50	225.00	14.40	3240.00	1.00	0.000
T O T A L S			225.00	3240.00			

Figure 30: DMAS B-52 scenario

Because there was no reasonable way to establish authorized quantities for the non-optimized (NOP) components in a timely manner for each aircraft-type and aircraft level, the kits were computed with NOP components in the stock computations although this is not a standard Air Force practice. The LRU pipeline quantities were computed to support 100% of the expected demands over the average repair cycle time. The availability goal was set at 80% for all runs. A sample of a DMAS component computation summary is presented below for the same B-52 run referenced above.

DMAS Version 3.3
Deployment Computation Output

Database: B52H Section: AAA

COMPUTATION SUMMARY

Data Section: AAA Date: 07-04-95 Time: 12:58

The following input parameters were specified for this report:

NFMC #1 Days: 1-90 Goal: 1.0 Percentage: 20.00%

100% of LRU Pipeline Bought
100% of SRU Pipeline Bought

The computed quantities support the following performance:

225 Sorties

3240 Flying Hours

	RR LRUs	RRR LRUs	SRUs	TOTAL
Line Items (Qty > 0)	15	0	0	15
Units	17	0	0	17
Cost (\$ Millions)	0.5	0.0	0.0	0.5
STOCK LEVELS				
0 Units	25	0	0	25
1 - 2 Units	14	0	0	14
3 - 5 Units	1	0	0	1
6 - 10 Units	0	0	0	0
11 - 20 Units	0	0	0	0
21 - 30 Units	0	0	0	0
31 - 40 Units	0	0	0	0
> 40 Units	0	0	0	0

Figure 31: Sample of DMAS computation summary

The DMAS program also provides a listing of the actual computed quantities. A partial sample of this listing is shown in the figure below:

Data Section: AAA

Date: 07-04-95

Time: 13:02

All NSNs in this part of the report show DMAS computed quantities for the DMAS EST QTY.

NFMC #1 Days: 1-90 Goal: 1.0 Percentage: 20.00%

NSN	WUC	NOMENCLATURE	PART TYPE	AUTH QTY	DMAS EST QTY
1280001866298	77DCA	ELECTRONIC	LC	7	0
1280011207217	73LH0	CONTROL, CO	LC	3	0
1280011513174	73QB0	CONVERTER	LC	2	0
1280012088417	75AAJ	INTERVALOM	LC	4	0
1280012270719	77EG0	CONTROL UN	LC	5	1
1280012283930	73AH0	RECEIVER-T	LC	11	1
1280012285349	73AAA	ELECTRONIC	LC	1	0
1560005926437	11DBD	DOOR, AIRCR	LC	2	0
1560006059700	14EBA	GEAR UNIT	LC	2	0
1560008976850	14AKA	SPOILER, WI	LC	4	0
1560008976853	14AKA	SPOILER, WI	LC	3	0
1560008976866	14AKA	FLAP, WING	LC	2	0
1560008978397	14AKA	FLAP, WING	LC	2	0
1560010598767	11EMA	DOOR, AIRCR	LC	2	0
1660004098966	41KAF	CONTROLBOX	LC	3	0
1660011591246	45FAB	VALVE ASSE	LC	3	0
1660011680330	41NAP	CONTROL BO	LC	3	0
4320006847073	46EAA	PUMP UNIT	LC	3	0
4810008125687	41GAZ	VALVE, BUTT	LC	7	1
5821012287058	63AK0	RECEIVER-T	LC	10	1
5841010781344	73MA0	RECEIVER-T	LC	5	0
5865010887202	76TB0	RECEIVER-T	LA	20	1
5895000395486	76WC0	INDICATOR	LC	17	1
5895009049816	76WE0	AMPLIFIER	LC	7	1
5895010620002	63GG0	MODEM, COMM	LC	1	0
5895010701444	63GH0	CONTROL, IN	LC	3	0
5895011327476	76NB0	DUPLEXER	LC	6	0
5895011525214	76EF0	CONTROL-IN	LC	4	0

6220011987217	44AFA	LIGHT, TAXI	LC	4	1
6340008117801	49DDA	CONTROL, AL	LC	6	1
6605012094229	73PCO	BATTERY AN	LN	18	3
6610005300026	51ACA	INDICATOR	LC	8	1
6610010456372	51ANK	INDICATOR	LC	8	1
6615005568510	51ANE	GYROSCOPE	LC	2	0
6615007220964	52EE0	CONTROLLER	LC	5	1
6615011655941	14FND	CONTROL UN	LC	9	0
6615012253746	52ED0	ELECTRONIC	LC	7	1
6625004713174	65BDA	TEST SET, T	LC	6	1
6680011462360	51BBK	INDICATOR	LC	3	0
6680011469527	51BBJ	INDICATOR	LC	4	0

NOTE: Part Type code for this output is:

1st character L=LRU S=SRU

2nd character N=NOP A=ADJ C=Computed U=No Data

3rd character R=RRR Blank=RR

Figure 32: Sample of DMAS Deployment Worksheet

Values in the column headed by DMAS EST QTY are the stock quantities of interest and represent the expected number of spares necessary to support 80% of the aircraft for the input scenario. These quantities were used as the "Full Stock" quantities in the research and simulation model runs in section 4.2.4.

As mentioned earlier, an exception had to be made for the random selection of components for the F-16. This was due to the high reliability of this aircraft. When the randomly generated input files were used, no spares were computed or necessary to support the aircraft. Therefore,

the top 20 failing items from the complete database were selected first and then the remaining components were selected randomly to complete the file. This method gave an input file that made spare quantities necessary to satisfy the availability requirements.

The "Partial Stock" quantities were established using the "Full Stock" quantities given above and the aircraft fill rate percentages given in Table 22. This was done in the following manner. First, the number of missing items was calculated and rounded to the nearest integer using the formula below:

$$MI = FSQ - FSQ * FP \quad (4-9)$$

where:

MI - Total number of missing spares

FSQ - Total number of spares in "Full Stock" case

FP - Aircraft Fill Rate Percentage

Then each spare in the "Full Stock" case was numbered from one to FSQ. Then MI unique random numbers which were less than FSQ were drawn from Table XII.4 Random Units in Beyer (1986). Those spares that matched the random numbers drawn became the missing items for the "Partial Stock" case. For example, using our B-52 case,

$$MI = 17 - 17 * 0.89 = 2$$

Thus, two items are missing from the B-52 package in this "Partial Stock" case. Two random numbers, 16 and 8, are drawn from the random number table and spare #8 and spare #16 will be deleted from the input data file. The "Zero Stock" case implies that all stock levels are set to zero.

After moving the DMAS computed quantities from DMAS into the aircraft databases (Note: the user-interface has a function that automates this process), both the research and simulation input files can be built for use in the research study. Now let's consider the steps necessary to analyze the simulation model.

4.2.3 Simulation Analysis Plan

The following is an outline of the procedure used to create the simulation results for comparison with the research model results.

Step 1: The measure of interest is a time-based steady-state statistic, therefore, run the model for a long time (Y units) and collect the time-based statistic every X time-units, giving a total of $Z = Y/X$ observations. The time-based statistics of interest are the availability of

all the aircraft operating and the availability of all the aircraft minus one operating.

Example: This was done initially, by running the model for 20000 (Y) days, collecting a time-based observation every 200 (X) days. This results in 100 (Z) observations for each measure.

Step 2: Since the system begins empty and idle and steady state results are desired, plot the time-sequenced observations to check for any transient effect that might occur early in the data that would bias the results. If transient effects are found the run length should be increased and the initial samples that are biased by the transient effect can be truncated.

Example: The figure below shows the graph of 100 observations of the simulation model. There is no apparent warm-up period in these results.

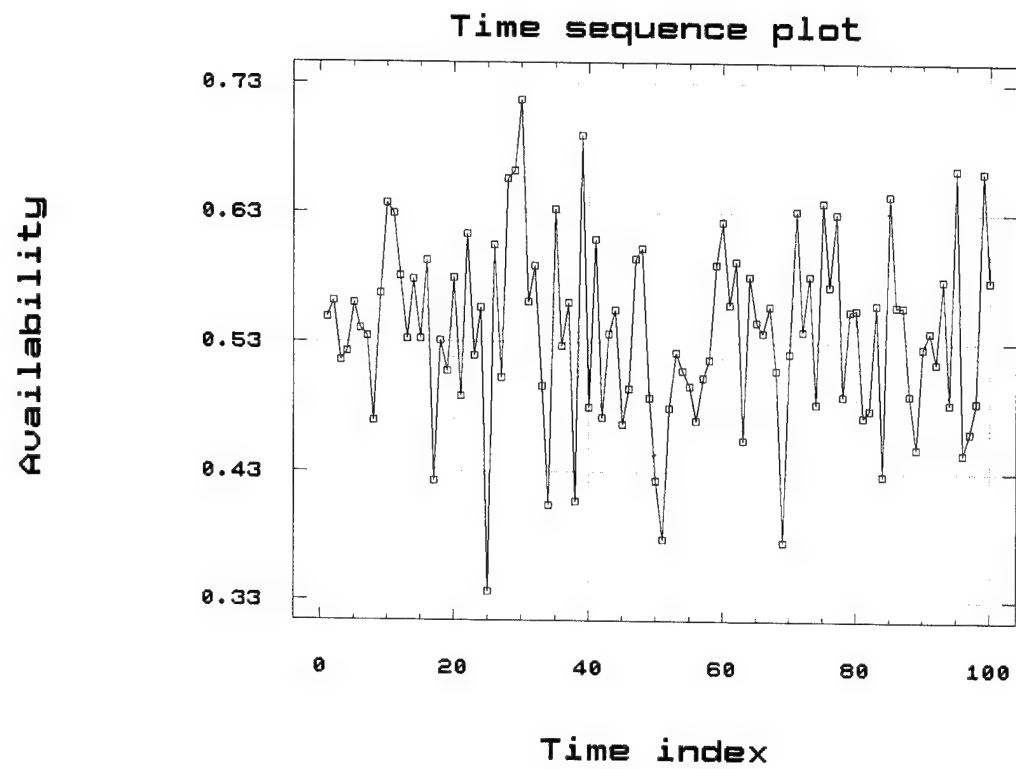


Figure 33: Time-based plot of simulation results

Step 3: Calculate the autocorrelations for the observations to determine if the sampling duration is long enough. If correlation exists between samples, the observation duration must be increased in order to reduce/eliminate the correlation between samples.

Example: The next figure shows a plot of the autocorrelations up to lag 10 of the availability of all the aircraft being up. There is no statistically significant correlation up to lag 10, therefore no additional runs need to be made.

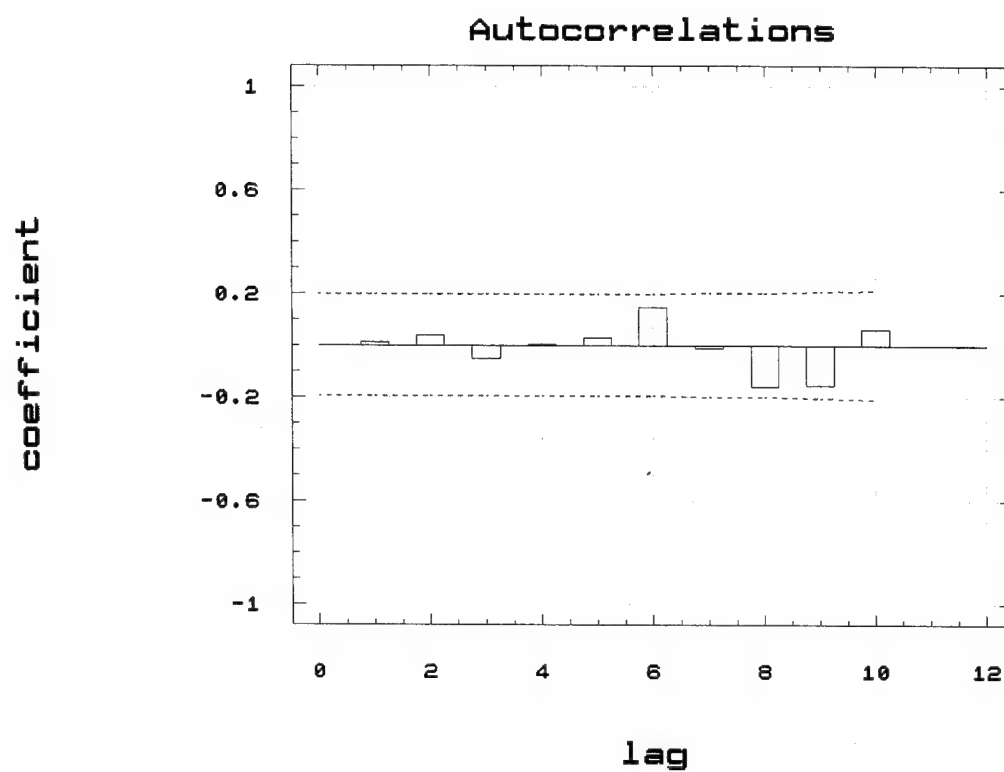


Figure 34: Plot of potential autocorrelation among the availability samples

If autocorrelation remains a problem, then a better estimate of the variance can be calculated using the autocorrelations from these runs. The technique is presented below and was taken from Banks and Carson (1984).

First, define the lag p covariance for a covariance stationary time series, X_1, X_2, \dots, X_p as

$$\gamma_p = \text{cov}(X_1, X_{1+p}) = \text{cov}(X_i, X_{i+p}) \quad (4-10)$$

where the value γ_p is independent of i by the definition of covariance stationary. For $p = 0$, the lag p covariance is the population variance (i.e. $\gamma_0 = \text{var}(X_i)$). The lag p autocorrelation is then equal to:

$$\rho_p = \frac{\gamma_p}{\gamma_0} \quad (4-11)$$

Based on the above and a sample mean $\hat{\mu}$ of n samples, the variance of $\hat{\mu}$ is:

$$\sigma^2(\hat{\mu}) = \text{var}\left(\frac{\sum_{i=1}^n X_i}{n}\right) = \frac{\gamma_0}{n} \left[1 + 2 \sum_{p=1}^n \left(1 - \frac{p}{n}\right) \rho_p\right] \quad (4-12)$$

The code to implement this procedure is shown in the figure below:

```

Program Main

integer i,k,auto
real sum,ele,mean,var,row(50),lower,upper,a
open(unit=7,file='eus.txt',status='old')
open(unit=8,file='eus.out',status='old')

read(7,*) auto
do 30 f=1,auto
sum=0
read(7,*) mean
read(7,*) var
write(8,*) '    Input mean & variance'
write(8,*)mean,var
write(8,*) ' '
print *,mean,var
do 10 i=1,50
read(7,*) k,row(i),a
10  continue
do 20 j=1,50
ele=(1.0-j/50.0)*row(j)
sum=sum+ele
c   print *,sum,ele,row(j),(1.0-j/50.0)
c   pause
20  continue
c   pause

covar=(var/50.0)*(1.0+2.0*sum)
print *, sum,covar
upper=2*sqrt(covar)+mean
lower=mean - 2*sqrt(covar)
print *,lower,mean,upper
write(8,*) '    New lower CB          Mean          Upper CB'
write(8,*)lower,mean,upper
write(8,*) ' '
write(8,*) ' '

30  continue

return
end

```

Figure 35: Code to calculate the sample variance using covariance

Once this variance estimate is made, a confidence interval can be built using the techniques from section 4.2.1.

Step 4: Calculate the standard deviation and confidence interval for the availability means and determine if the desired accuracy has been achieved. If yes, then use these results for comparisons with the research model. If no, determine the number of additional runs necessary based on the current achieved sample variance and repeat the previous steps. Although the military requirements for accuracy are only $\pm 20\%$ (see sec 4.2.1), this research will use a much smaller interval as a discriminator. This research will use $\pm 2.5\%$ as a goal for accuracy.

Example: The mean for the E-3B availability of 5 aircraft is 19.34 with a standard deviation of 0.0604 which gives a 95% confidence interval of 18.14% to 20.54%. This is much smaller than the required interval of $\pm 2.5\%$

The steps just presented will provide the results necessary for comparison with the research model. Next, the model outputs resulting from the application of the steps in the last two sections will be compared.

4.2.4 Research Model And Simulation Output Compared

The comparison of the research method to the simulated "live" results will be broken into two parts. First, the results for the no component redundancy case will be presented and then the results for the component redundancy case will be given.

4.2.4.1 No Component Redundancy Case

Both the simulation and the research model were run using duplicate scenario and component data as described earlier in section 4.2.1 and 4.2.2. The results presented below will be for the non-redundancy case. The figure below shows a sample of the research model output for the non-redundancy case.

THIS IS THE NUMBER OF AIRCRAFT 3
 THIS IS THE NUMBER OF LRUs ANALYZED 40
 THIS IS THE NUMBER OF SRUs ANALYZED 0

THIS IS THE PROBABILITY OF ALL SYSTEMS UP - 39.6446249415E-02
 THIS IS THE PROBABILITY OF 2 SYSTEMS UP - 14.3070082870E-02

This run had 22 cycles through the parts.

The availability of 3 systems is 39.6446249415E+00
 The availability of 2 or more systems is 82.5656498024E+00
 The availability of exactly 2 systems is 42.9210248609E+00

This is a list of the top 5 problem parts

Stock Number	E[O]
5895012511203	2.820E+00
5821011095573	2.868E+00
6615012107853	2.901E+00
5841010797775	2.933E+00
1660012756785	2.958E+00

Figure 36: E-3B partial stock research model output

Below is a partial listing of the simulation output for the same case presented above. This figure only shows the first 20 rows of output. The actual output contains 100 rows.

1.98483	0.704144	0.330639	42	32
1.88078	0.687282	0.249058	49	40
2.36169	0.893590	0.482015	26	25
2.36886	0.908599	0.460260	36	33

2.16649	0.829668	0.338184	32	30
2.35466	0.855867	0.510129	32	26
2.04268	0.761439	0.297364	41	30
2.05337	0.735016	0.335941	45	39
2.34288	0.887205	0.455680	30	30
2.07796	0.760852	0.347483	34	30
2.15588	0.823139	0.359085	37	29
2.18172	0.809198	0.381926	44	39
2.19557	0.836151	0.373369	39	31
2.23828	0.900003	0.342131	28	27
2.08283	0.812630	0.273210	31	29
2.17627	0.794334	0.407557	32	26
2.07389	0.779984	0.335586	42	36
2.37551	0.884810	0.501597	29	26
2.18475	0.814333	0.386766	41	32
2.28828	0.818224	0.500368	28	20

Figure 37: E-3B partial stock simulation model output

The following is a description of each column in this output:

- 1) The average aircraft availability
- 2) The average percentage of time two or more aircraft
(M-1) were available
- 3) The average percentage of time all three aircraft
(M) were available

- 4) Number of times that the status of availability of M-1 or more aircraft is checked as less than M-1 aircraft up
- 5) Number of times that the status of availability of M-1 or more aircraft changes from less than M-1 aircraft up to at least M-1 aircraft up

The initial simulation runs and analysis resulted in the need for several reruns of the model due to the presence of some transient effects and some high autocorrelations among the output variables of interest. An autocorrelation was considered significant if it was greater than 0.2 and occurred at lag 10 or less. The following series of figures gives a sample of results with and without transient effects and then a sample with high autocorrelation problems and one sample without.

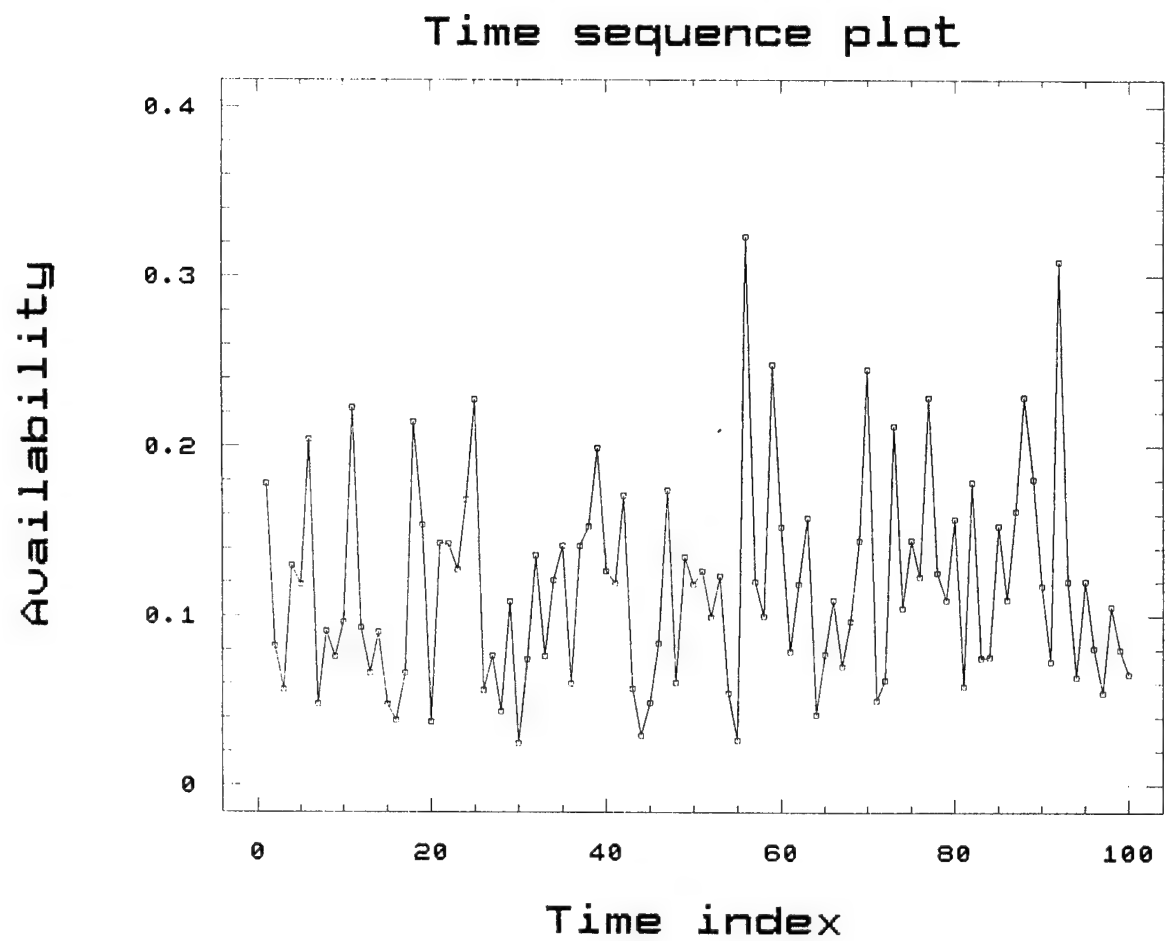


Figure 38: Time series plot of simulation results without
transient effect - F-16C run

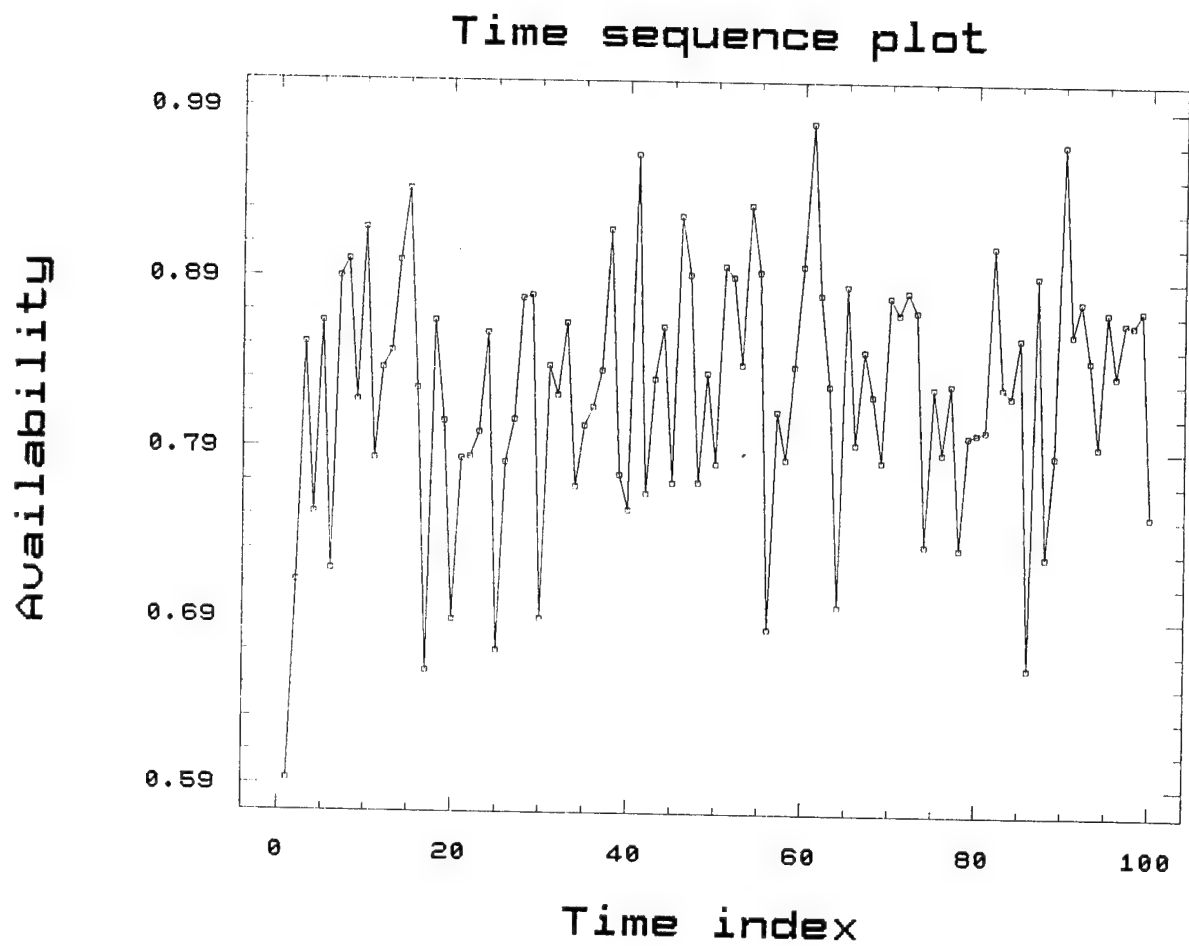


Figure 39: Time series plot of simulation results with
transient - E-3B run

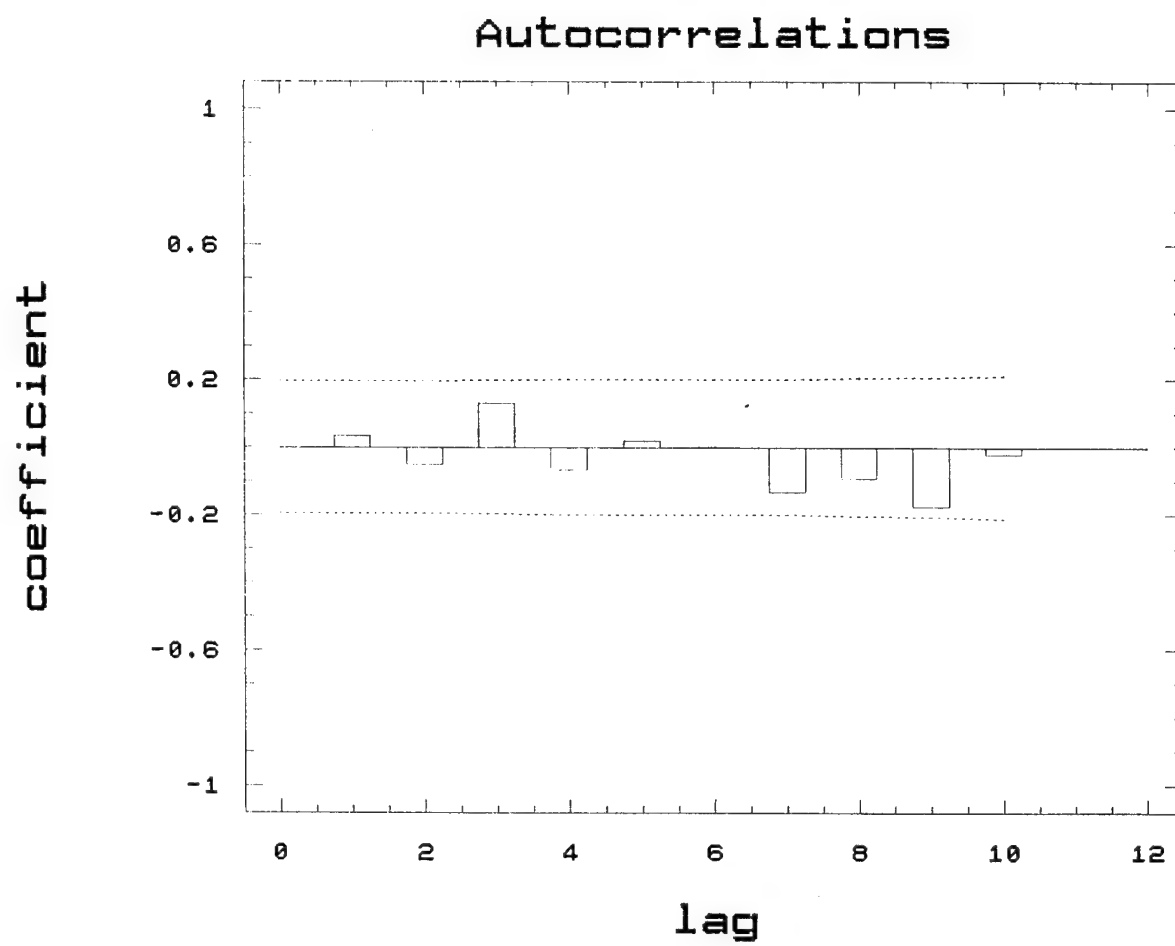


Figure 40: Plot of insignificant simulation
autocorrelations - F-16C run

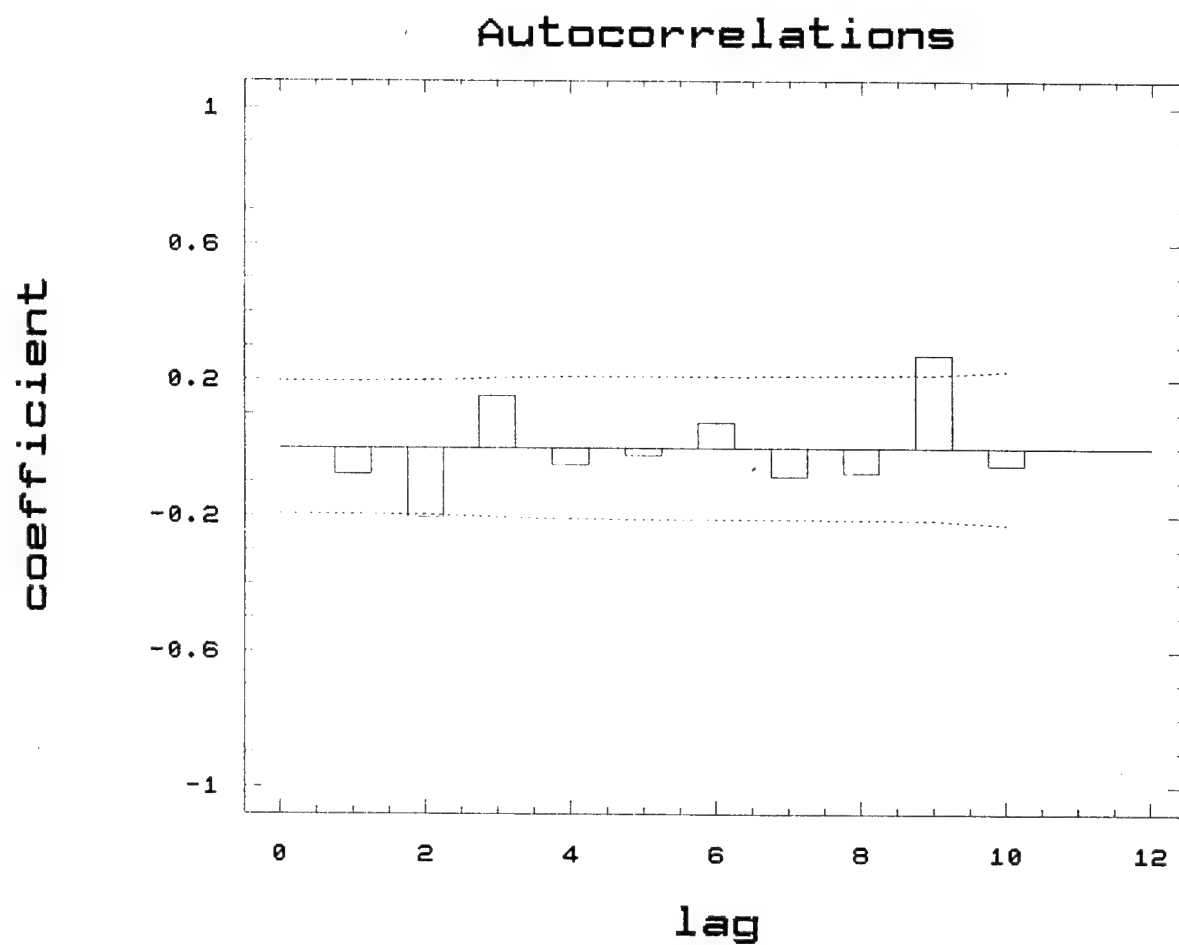


Figure 41: Plot of high simulation autocorrelations - F-15C
run

The table below is a listing of the aircraft type, scenario, and reasons for rerunning the model for each sample.

Table 24: No Redundancy Transient and Autocorrelation

Reruns

Aircraft	No. of A/C	Stock	Reason	Variable*	Correlation Value
B52	1	Partial	Transient	Avail 1	
B52	3	Zero	Lag 9	Avail 3	0.2284
			Lag 10	Avail 3	0.2583
B52	3	Full	Lag 2	Avail 2	0.2947
			Lag 3	Avail 3	0.2801
B52	5	Full	Lag 2	Avail 5	0.3769
E3B	5	Zero	Lag 3	Avail 4	0.2221
E3B	5	Partial	Lag 1	Avail 4	0.2003
E3B	1	Full	Lag 4	Avail 1	0.2056
			Transient		
E3B	3	Full	Lag 8	Avail 2	0.2281
F15	5	Zero	Lag 1	Avail 5	0.2133
			Lag 7	Avail 5	0.2421
F15	5	Partial	Lag 4	Avail 4	0.3188
			Transient	Avail 4	
F15	5	Full	Lag 2	Avail 5	0.2046
			Lag 9	Avail 5	0.2717
F16	5	Zero	Lag 2	Avail 5	0.2120
F16	5	Full	Lag 8	Avail 4	0.2179
			Lag 8	Avail 5	0.2171

* Avail # - This is the variable name for the availability of # aircraft.

As a result of these problems, the sample length, X , was doubled to 400 days to reduce the correlation between samples and the first five samples were discarded to eliminate the transient problems. These changes resulted in a total run length, Z , of 1,008,000 hours or 42,000 days. The new runs were made and the data analyzed. The figures below show how the new run length parameters eliminate the transient data and reduce the autocorrelation among the output data. These figures correlate with the examples given earlier in this section.

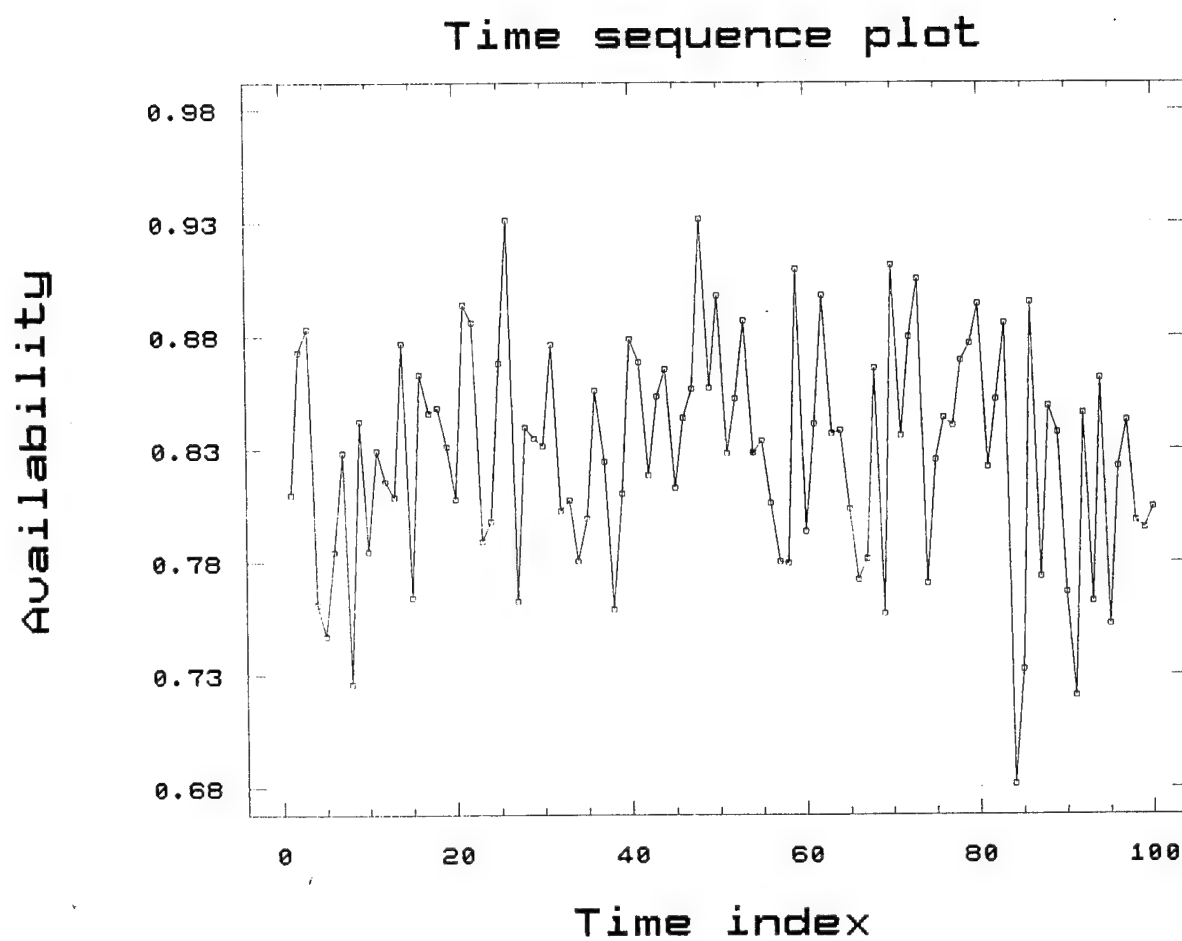


Figure 42: Elimination of transient data - E-3B run

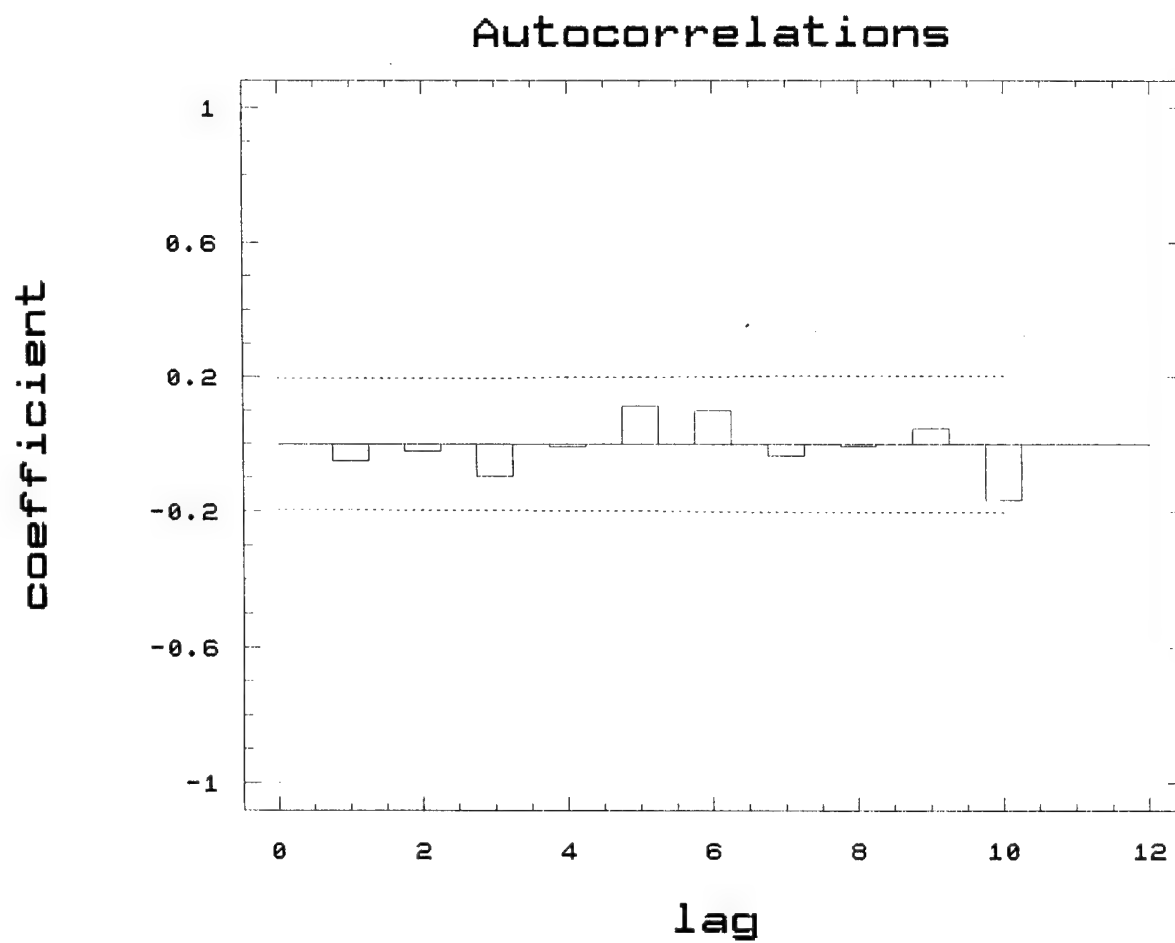


Figure 43: Reduction of autocorrelation - F-15C run

These new runs completely eliminated the transient problem but they did not eliminate all of the correlation problems. The table below presents the remaining autocorrelation problems.

Table 25: No Redundancy Transient and Autocorrelation
Reruns

Aircraft	No. of A/C	Stock	Reason	Variable*	Correlation Value
B52	1	Partial	Lag 1	Avail 1	0.2366
			Lag 4	Avail 1	0.2947
B52	3	Full	Lag 2	Avail 2	0.2284
			Lag 4	Avail 2	0.2317
B52	5	Full	Lag 1	Avail 5	0.2346
			Lag 3	Avail 5	0.2103
E3B	5	Zero	Lag 1	Avail 4	0.2221
			Lag 2	Avail 5	0.2094
E3B	3	Full	Lag 1	Avail 3	0.2531
F15	5	Zero	Lag 1	Avail 5	0.2201
F15	5	Partial	Lag 6	Avail 4	0.2029
F16	5	Zero	Lag 2	Avail 4	0.2903
			Lag 6	Avail 4	0.2259

Rather than continue to rerun the simulation model for these cases and attempt to eliminate the significant autocorrelation, a different estimate of the variance of the mean was calculated using the autocorrelations from these runs. This variance estimation technique was presented in section 4.2.3. Once this variance estimate is made, a confidence interval can be built using the techniques from section 4.2.1. Using the results from the research model and the simulation model, the tables below present the validation results for each aircraft and stock level case.

Table 26: No Redundancy Comparison of Results - Full Stock
Case

	# OF A/C	RESEARCH	MODEL	Simulation 95% C.I.		
		MODEL	DIFFERENCES	LOWER	MEAN	UPPER
E3B	5	19.40	0.06	18.14	19.34	20.54
		37.50	-0.06	36.49	37.56	38.62
	3	48.28	0.07	46.91	48.21	49.51
		39.71	0.06	38.77	39.65	40.53
	1	82.38	-0.43	81.84	82.81	83.79
F15	5	24.33	0.99	22.35	23.34	24.33
		39.67	0.22	38.66	39.45	40.24
	3	53.90	1.18	50.99	52.72	54.45
		36.94	-1.09	36.52	38.03	39.53
	1	86.89	-0.23	85.70	87.12	88.54
B52	5	25.78	-0.33	25.34	26.11	26.89
		37.97*	-0.73	38.20	38.70	39.20
	3	54.34	0.07	53.29	54.27	55.25
		35.67	-0.60	35.65	36.27	36.89
	1	82.38	-0.09	81.45	82.47	83.49
F16	5	87.97	-0.42	87.49	88.39	89.29
		11.41	0.48	10.11	10.93	11.75
	3	94.76	-0.46	94.50	95.22	95.93
		5.142	0.459	3.997	4.683	5.368
	1	97.73	0.27	96.89	97.46	98.04

* Outside of the simulation 95% confidence interval.

Table 27: No Redundancy Comparison of Results - Partial
Stock Case

	# OF A/C	RESEARCH	MODEL	Simulation 95% C.I.		
		MODEL	DIFFERENCES	LOWER	MEAN	UPPER
E3B	5	15.23	-0.13	14.60	15.36	16.13
		34.74	0.25	33.67	34.49	35.31
	3	39.27	-0.33	37.87	39.60	41.32
		43.02	0.61	41.24	42.41	43.58
	1	77.09	-0.14	75.79	77.23	78.66
F15	5	19.17	0.63	17.17	18.54	19.40
		37.47	0.02	36.91	37.45	37.99
	3	44.22	1.02	41.56	43.20	44.84
		41.44	-0.12	40.22	41.56	42.89
	1	78.42	1.01	76.04	77.41	78.78
B52	5	14.39	0.49	13.10	13.90	14.71
		33.66	0.23	32.48	33.43	34.38
	3	51.99	0.01	50.63	51.98	53.34
		36.95	-0.4	36.35	37.35	38.36
	1	80.36	0.01	79.19	80.35	81.51
F16	5	76.48	0.38	74.53	76.10	77.66
		20.91	-0.01	19.57	20.92	22.27
	3	85.19	-0.98	84.86	86.17	87.48
		14.01	0.85	11.94	13.16	14.38
	1	95.01	0.01	94.20	95.00	95.79

Table 28: No Redundancy Comparison of Results - Zero Stock
Case

	# OF A/C	RESEARCH	MODEL	Simulation 95% C.I.		
		MODEL	DIFFERENCES	LOWER	MEAN	UPPER
E3B	5	8.409	-0.138	8.03	8.547	9.064
		26.94	-0.22	26.21	27.16	28.12
	3	22.85	-0.12	21.65	22.97	24.29
		43.58	-0.42	42.81	44.00	45.20
	1	61.14	0.66	59.01	60.48	61.95
F15	5	5.284*	-0.594	5.608	5.878	6.148
		21.15	-0.63	21.11	21.78	22.46
	3	17.77	0.10	16.55	17.67	18.79
		41.51	-0.43	40.93	41.94	42.94
	1	56.31	0.95	53.91	55.36	56.81
B52	5	2.823*	-0.263	2.850	3.086	3.323
		14.70	-0.45	14.53	15.15	15.76
	3	12.13	-0.04	11.76	12.17	12.57
		37.12	-0.06	36.64	37.18	37.72
	1	49.59	-0.42	48.81	50.01	51.21
F16	5	11.96	0.03	11.34	11.93	12.53
		31.65	-0.14	30.75	31.79	32.83
	3	27.97	0.04	26.55	27.93	29.31
		44.40	-0.47	43.77	44.87	45.97
	1	65.40	0.79	63.10	64.61	66.12

* Outside of the simulation 95% confidence interval

The research results marked with an asterisk are the research model predictions that do not fall within the simulation 95% confidence intervals. Fifty seven of the sixty research model results (95%) lie within the simulation 95% confidence intervals. Below is a histogram plot of these differences.

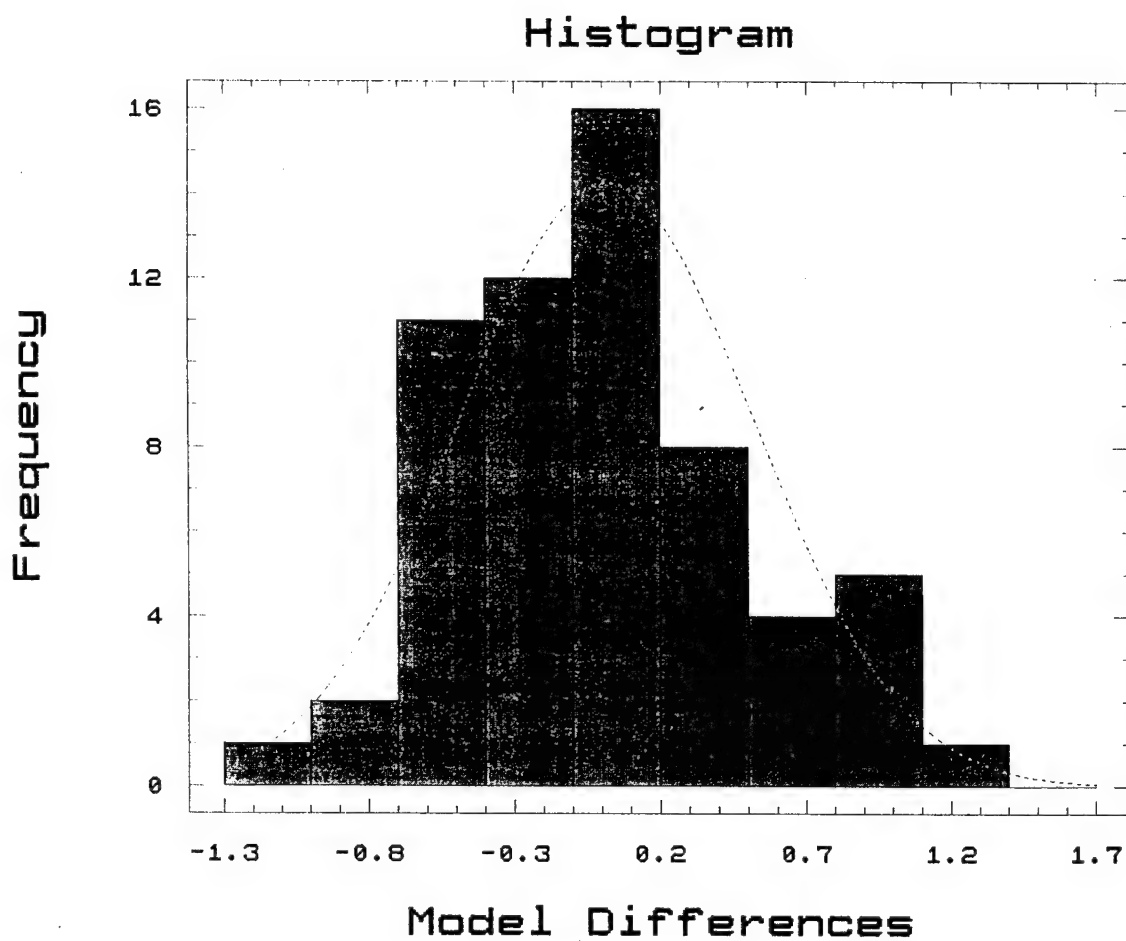


Figure 44: Histogram of no component redundancy model
differences

An analysis of the model differences provides the following statistics output from Execustat (Strategy Plus, 1993):

```
-----  
Sample size = 60  
Mean = 0.0167333  
Variance = 0.253504  
Std. deviation = 0.503492  
  
95% confidence intervals  
Mean: (-0.113333,0.146799)  
Variance: (0.182132,0.377182)  
Std. deviation: (0.426769,0.614151)  
-----
```

Figure 45: Sample mean and variance of differences - no component redundancy case

The confidence intervals and the hypothesis means test assume that the sample is from a normal distribution. A normal probability plot of the differences is presented below:

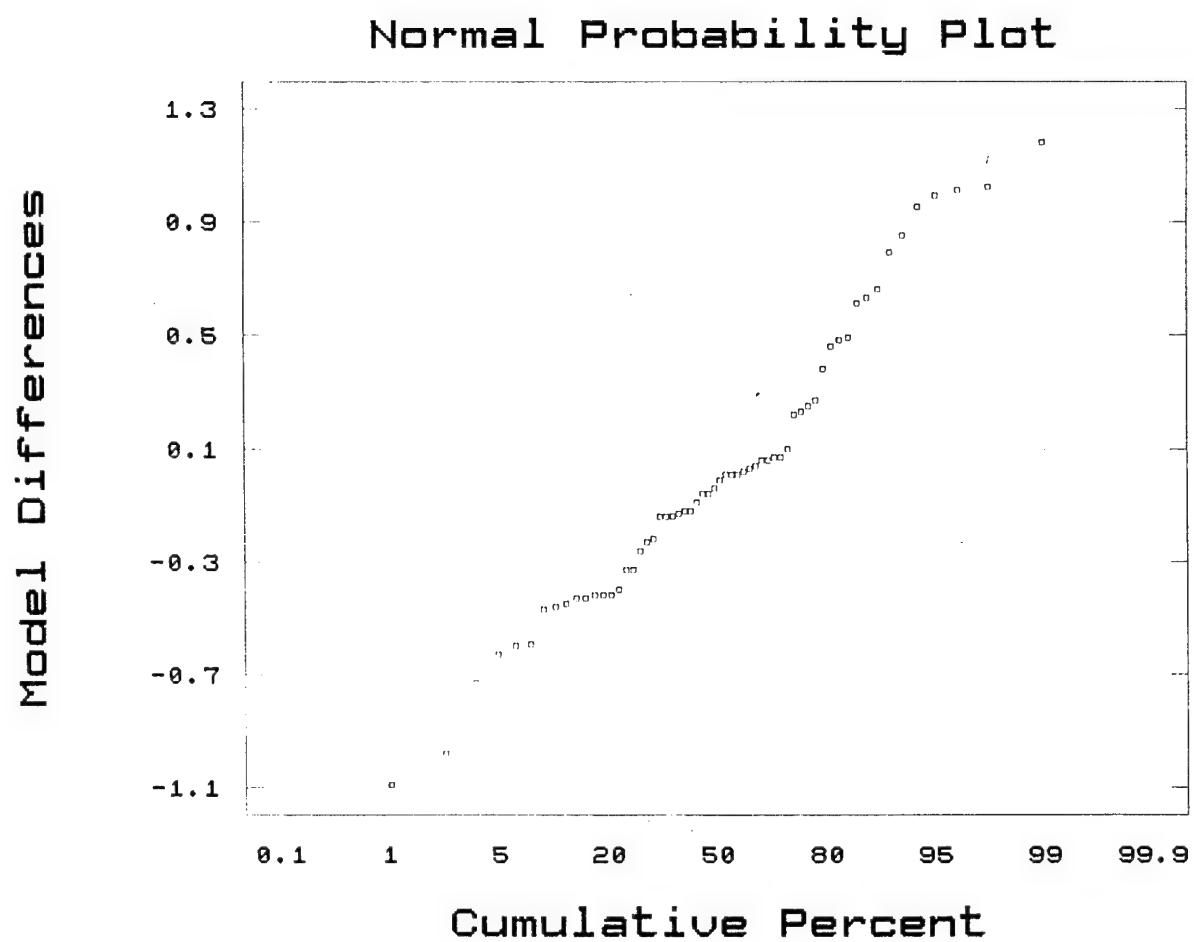


Figure 46: Normal probability plot - no redundancy case

To test the hypothesis that the sample is from a normal distribution, a Shapiro-Wilks test was used. This test was discussed earlier in section 4.2.1. Using Execustat (Strategy Plus, 1993), the test statistic computed by the program was 0.959729 and the P-value was 0.1017. This implies that at an $\alpha = 0.05$, the null hypothesis that the sample is from a normal distribution cannot be rejected.

Now the test of hypothesis on the model differences can be performed. The test is shown below:

Let $D_i = (\text{research output}_i - \text{simulation output}_i)$ and \bar{D} and S^2 be the sample mean and variance, then the two-sided test is:

$$H_0: \bar{D} = 0$$

$$H_a: \bar{D} \neq 0$$

with the following test statistic:

$$t_0 = \frac{\bar{D} - 0}{0.503492/\sqrt{60}}$$

This test statistic follows a t distribution with 59 degrees of freedom if H_0 is true. After t_0 is calculated, H_0 is rejected if either $t_0 > 2.00$ or $t_0 < -2.00$ where 2.00 and -2.00 are the upper and lower 0.025 percentage points

of the t distribution with 59 degrees of freedom. Execustat gave the following t -statistic of 0.257434 and P -value of 0.7977. Based on an α value of 0.05, the null hypothesis that the model differences are equal to zero cannot be rejected. The power of the test is very high (β very small) using a difference value of 2.5%. The c - value is 4.966 which is off the OC charts in Hines and Montgomery (1980). The research model appears to perform well in the no component redundancy case but what about the component redundancy case.

4.2.4.2 Component Redundancy Case

Both the simulation and the research model were run using duplicate scenario and component data as described earlier in section 4.2.1 and 4.2.2. The results presented below will be for the component redundancy case. The figure below shows a sample of the research model output for the redundancy case.

THIS IS THE NUMBER OF AIRCRAFT 3
 THIS IS THE NUMBER OF LRUs ANALYZED 40
 THIS IS THE NUMBER OF SRUs ANALYZED 0

This run had 2 cycles through the parts.

The availability of 3 systems is 94.7742376230E+00
 The availability of 2 or more systems is 99.9048388481E+00
 The availability of exactly 2 systems is 51.3060122512E-01

This is a list of the top 5 problem parts

Stock Number	E[O]
5865011106043	2.977E+00
1005010446174	2.987E+00
4320013783398	2.995E+00
2925011150306	2.997E+00
2835013083769	2.997E+00

Figure 47: F-16C full stock research model output

Below is a partial listing of the simulation output for the same case presented above. This figure only shows the first 20 rows of output. The actual output contains 100 rows.

2.86517	0.995628	0.869542	12	12
2.95147	1.00000	0.951474	0	0
2.98936	1.00000	0.989357	2	2
2.98235	1.00000	0.982346	2	2
2.97390	0.996799	0.977105	1	1
2.88742	1.00000	0.887420	8	8

2.95395	1.00000	0.953949	5	5
2.97506	1.00000	0.975060	2	2
2.86948	0.988240	0.881240	10	10
2.98547	1.00000	0.985474	0	0
2.97803	1.00000	0.978029	2	2
2.99689	1.00000	0.996895	0	0
2.92470	1.00000	0.924696	3	3
2.96711	1.00000	0.967113	5	5
2.94679	1.00000	0.946792	1	1
2.96737	1.00000	0.967370	3	3
2.92542	1.00000	0.925423	4	4
2.91955	0.997957	0.921595	4	4
2.91914	1.00000	0.919139	4	4
2.94836	1.00000	0.948363	2	2

Figure 48: F-16C full stock simulation model output

The following is a description of each column in this output:

- 1) The average aircraft availability
- 2) The average percentage of time two or more aircraft
(M-1) were available
- 3) The average percentage of time all three aircraft
(M) were available

- 4) Number of times that the status of availability of M-1 or more aircraft is checked as less than M-1 aircraft up
- 5) Number of times that the status of availability of M-1 or more aircraft changes from less than M-1 aircraft up to at least M-1 aircraft up

The initial simulation runs and analysis resulted in the need for several reruns of the model due to the presence of some transient effects and some high autocorrelations among the output variables of interest. An autocorrelation was considered significant if it was greater than 0.2 and occurred at lag 10 or less. The following series of figures gives a sample of results with and without transient effects and then a sample with high autocorrelation problems and one sample without.

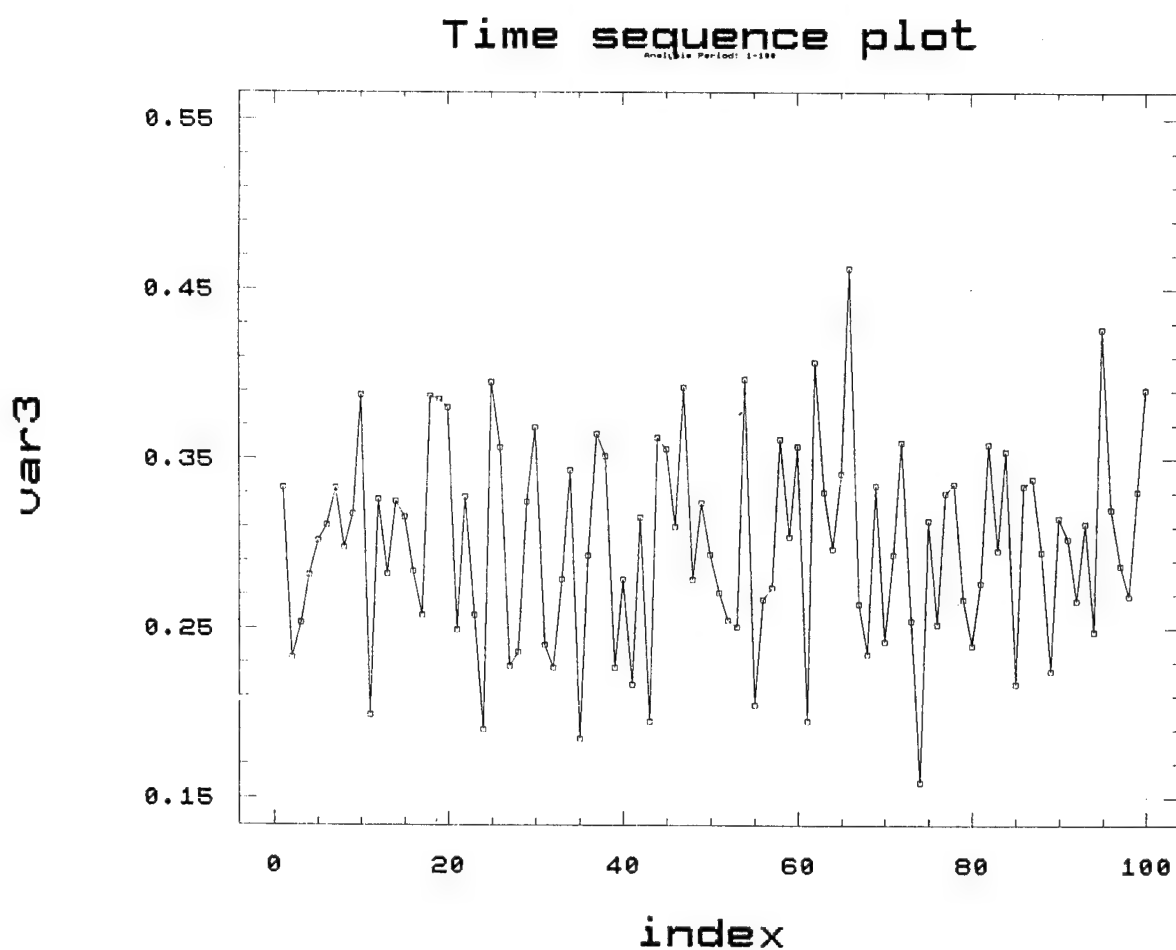


Figure 49: Time series plot of simulation results without
transient - B-52H run

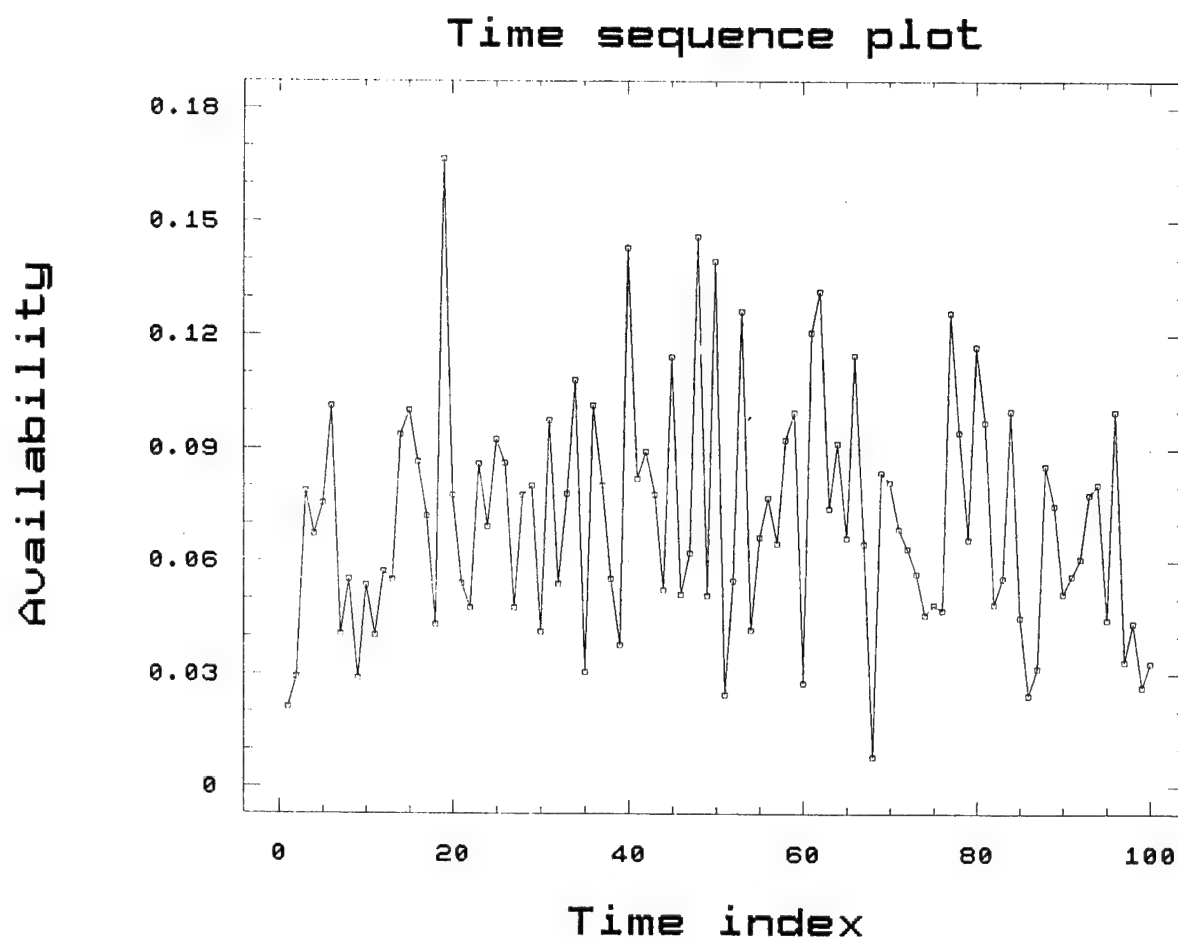


Figure 50: Time series plot of simulation results with
transient - F-15C run

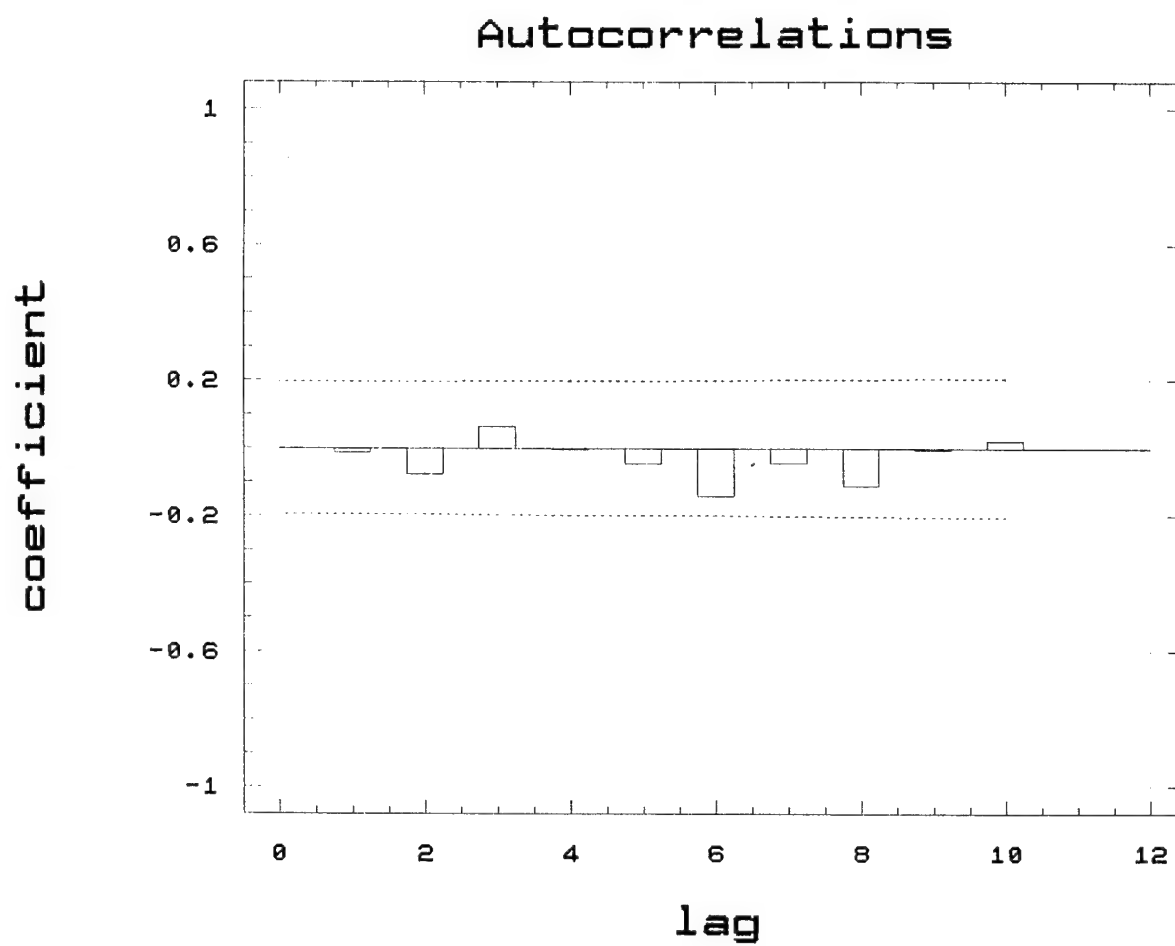


Figure 51: Plot of insignificant simulation
autocorrelations - F-15C run

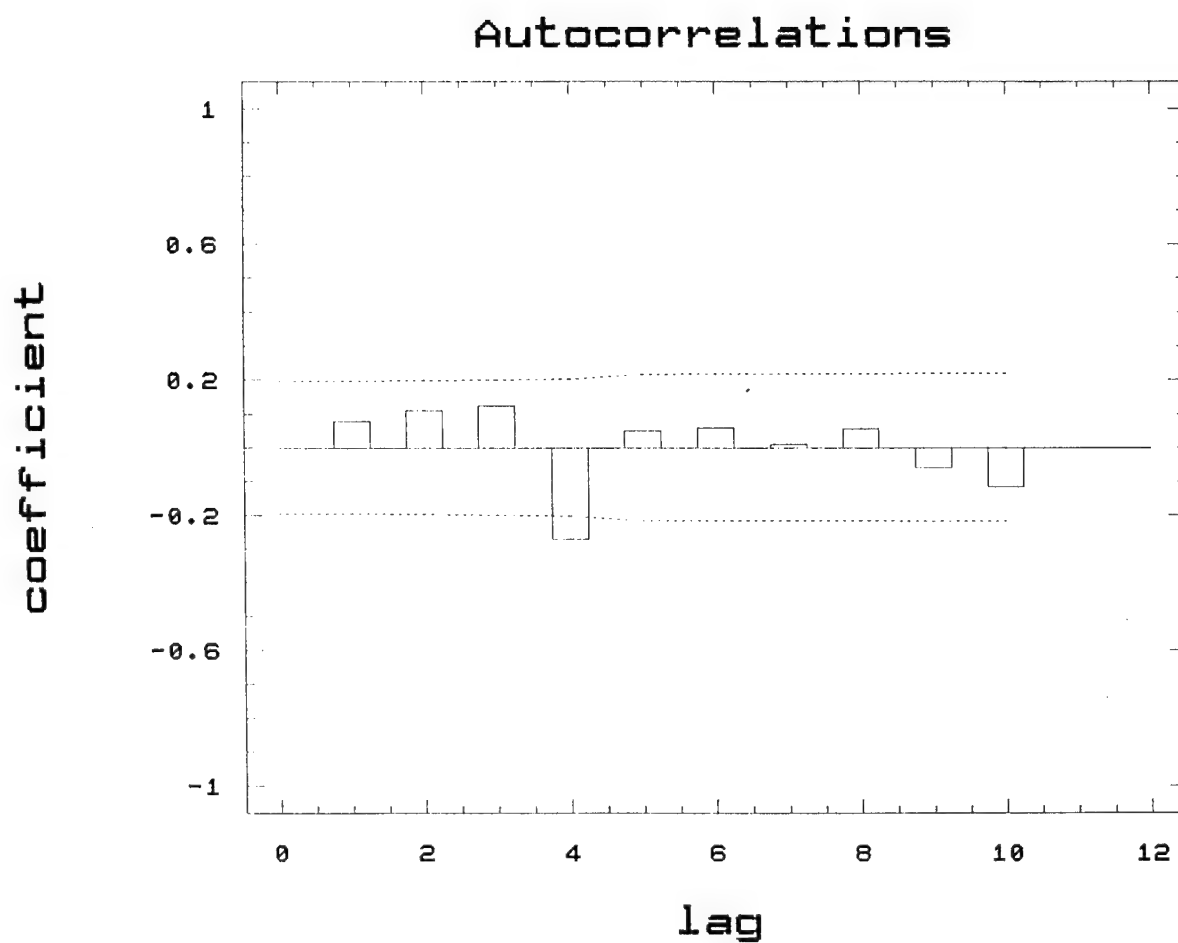


Figure 52: Plot of high simulation autocorrelations - F-15C
run

The table that follows is a listing of the aircraft-type, scenario, and reasons for rerunning the model for each sample.

Table 29: Redundancy Case Transient and Autocorrelation

Reruns

Aircraft	No. of A/C	Stock	Reason	Variable*	Correlation Value
B52	3	Zero	Lag 4	Avail 3	0.2025
			Lag 8	Avail 3	0.2256
B52	5	Zero	Lag 5	Avail 4	0.3195
E3B	5	Zero	Lag 2	Avail 5	0.3236
E3B	5	Partial	Lag 8	Avail 5	0.2317
E3B	1	Zero	Lag 3	Avail 1	0.2553
E3B	3	Zero	Lag 1	Avail 2	0.2537
F15	3	Zero	Lag 2	Avail 2	0.2383
F15	5	Zero	Transient	*****	
F15	5	Partial	Lag 4	Avail 5	0.2723
F15	5	Full	Lag 9	Avail 5	0.2196
F16	3	Partial	Lag 3	Avail 4	0.2479
			Lag 3	Avail 5	0.2126
F16	5	Zero	Lag 1	Avail 4	0.2503
F16	5	Full	Lag 8	Avail 4	0.2311
			Lag 8	Avail 5	0.2255

* Avail # - This is the variable name for the availability of # aircraft.

As a result of these problems, the sample length, X , was doubled to 400 days to reduce the correlation between samples and the first five sample were discarded to eliminate the transient problems. These changes resulted in a total run length, Z , of 1,008,000 hours or 42,000 days. The new runs were made and the data analyzed. The figures below show how the new runtime parameters eliminate the transient data and reduce the autocorrelation among the output data. These figures correlate with the examples given earlier in this section.

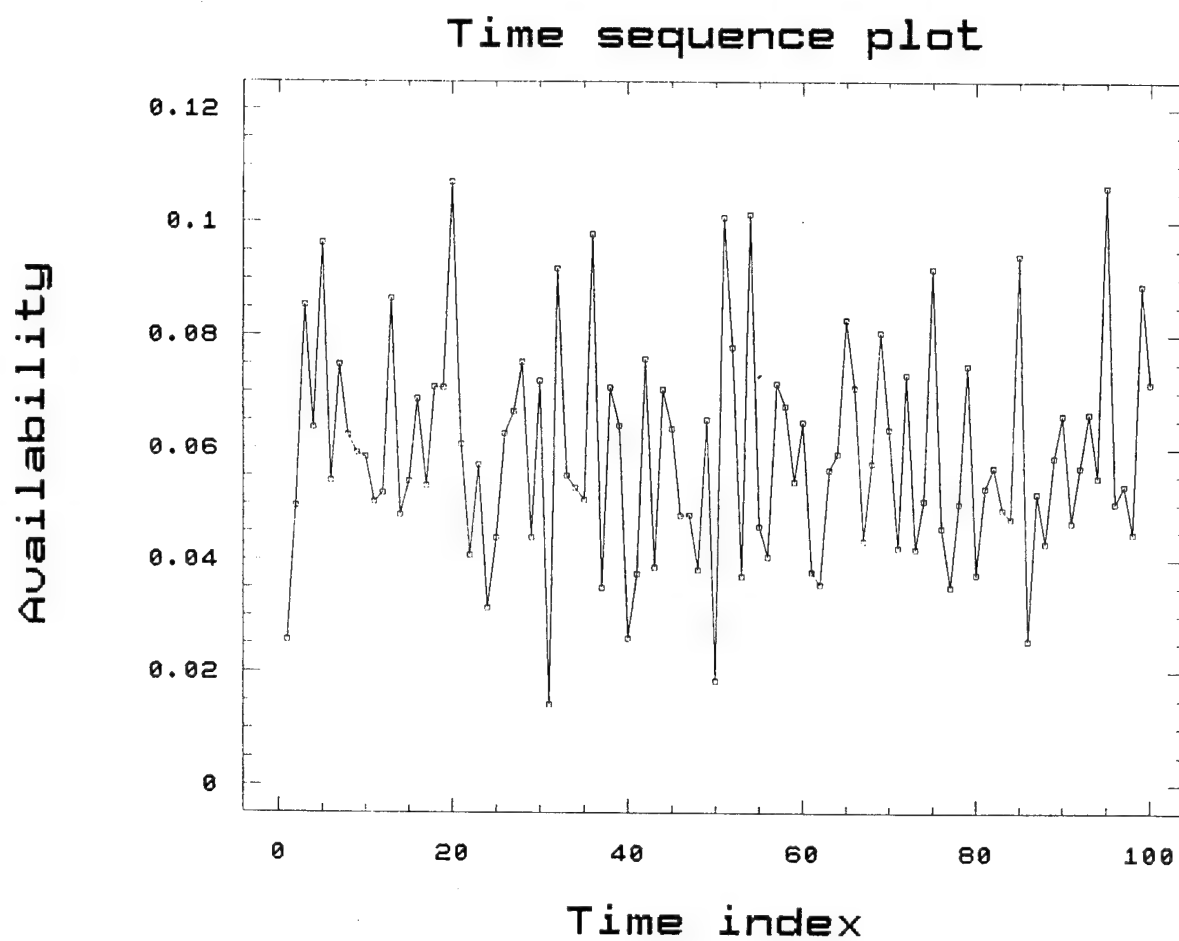


Figure 53: Elimination of transient data - F-15C run

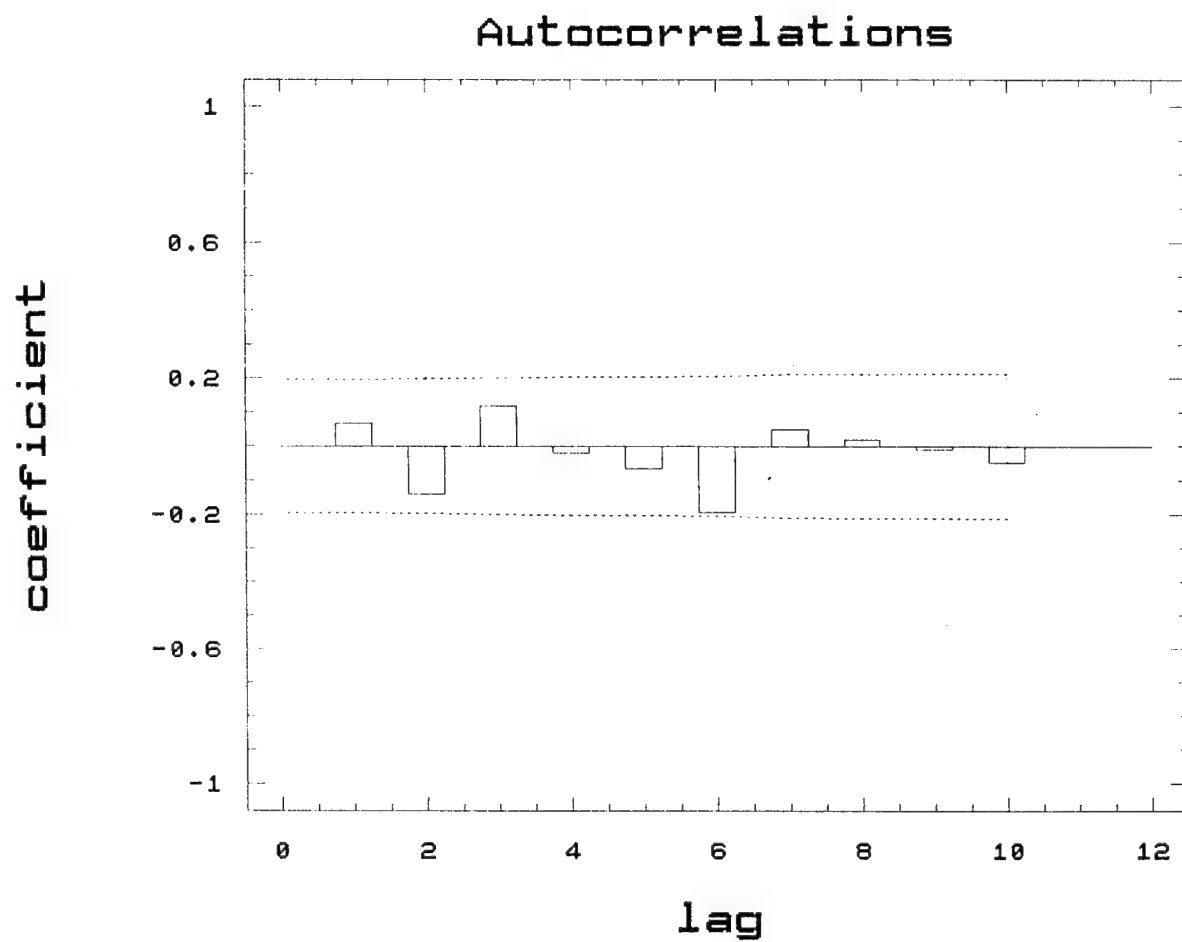


Figure 54: Reduction of autocorrelation - F-15C run

These new runs completely eliminated the transient problem but they did not eliminate all of the correlation problems. The table below gives the remaining autocorrelation problems.

Table 30: Redundancy Case Transient and Autocorrelation
Reruns

Aircraft	No. of A/C	Stock	Reason	Variable*	Correlation Value
B52	3	Zero	Lag 2	Avail 3	0.2710
B52	5	Zero	Lag 9	Avail 5	0.2461
E3B	5	Zero	Lag 9	Avail 5	0.2671
F15	3	Zero	Lag 1	Avail 2	0.2624
			Lag 1	Avail 3	0.2160
F16	5	Zero	Lag 2	Avail 4	0.2030
			Lag 3	Avail 5	0.2161

Rather than continue to rerun the simulation model for these cases and attempt to eliminate the significant autocorrelation, a different estimate of the variance of the mean was calculated using the autocorrelations from these runs. The technique was presented in section 4.2.3.

Once this variance estimate is made, a confidence interval can be built using the techniques from section 4.2.1.

Using the results from the research model and the simulation model, the tables below present the validation results for each aircraft stock level case.

Table 31: Redundancy Case Comparison of Results - Full
Stock Case

	# OF A/C	RESEARCH	MODEL	Simulation 95% C.I.		
		MODEL	DIFFERENCES	LOWER	MEAN	UPPER
E3B	5	22.61	-0.12	21.4	22.73	22.73
		39	0.47	37.49	38.53	39.58
	3	49.53	-0.35	48.25	49.88	51.5
		39.14	0.8	37.14	38.34	39.53
	1	82.99	0.07	81.5	82.92	84.33
F15	5	24.59	1.01	22.53	23.58	24.62
		39.75	0.2	38.79	39.55	40.32
	3	54.14	1.17	51.3	52.97	54.64
		36.81	-1.03	36.4	37.84	39.29
	1	86.94	-0.22	85.74	87.16	88.57
B52	5	29.65	-0.32	28.79	29.97	31.15
		38.38	-0.47	38.14	38.85	39.55
	3	58.78	0.39	56.98	58.39	59.81
		33.02	-0.67	32.59	33.69	34.78
	1	84.78	-0.27	84.19	85.05	85.91
F16	5	88.02	-0.43	87.55	88.45	89.35
		11.36	0.48	10.05	10.88	11.7
	3	94.77	-0.46	94.51	95.23	95.95
		5.131	0.461	3.981	4.67	5.359
	1	97.77	0.21	97	97.56	98.12

Table 32: Redundancy Case Comparison of Results - Partial
Stock Case

	# OF A/C	RESEARCH	MODEL	Simulation 95% C.I.		
		MODEL	DIFFERENCES	LOWER	MEAN	UPPER
E3B	5	17.67	0.07	16.81	17.6	18.38
		36.55	0.34	35.49	36.21	36.93
	3	45.47	0.26	43.58	45.21	46.85
		40.93	0.44	39.19	40.49	41.79
	1	77.63	-0.04	76.3	77.67	79.04
F15	5	19.36	0.54	17.96	18.82	19.67
		37.58	0.1	36.67	37.48	38.28
	3	46.46	0.71	43.96	45.75	47.54
		40.55	-0.13	39.21	40.68	42.16
	1	78.46	1.01	76.06	77.45	78.94
B52	5	16.31	0.46	14.96	15.85	16.74
		35.19	0.26	33.97	34.93	35.89
	3	56.18	-0.2	55.05	56.38	57.7
		34.61	-0.35	33.94	34.96	35.98
	1	82.72	-0.3	82.03	83.02	84.01
F16	5	76.52	0.31	74.64	76.21	77.77
		20.88	0.08	19.46	20.8	22.15
	3	85.56	-0.48	85.05	86.04	87.04
		13.69	0.42	12.33	13.27	14.21
	1	95.49	0	94.69	95.49	96.29

Table 33: Redundancy Case Comparison of Results - Zero
Stock Case

	# OF A/C	RESEARCH	MODEL	Simulation 95% C.I.		
		MODEL	DIFFERENCES	LOWER	MEAN	UPPER
E3B	5	11.13	0.17	10.6	10.96	11.33
		30.69	0.06	29.91	30.63	31.36
	3	27.1	0.56	25.55	26.54	27.54
		44.33	-0.35	43.96	44.68	45.41
	1	64.57	-0.34	63.84	64.91	65.97
F15	5	7.223	-0.158	6.905	7.381	7.858
		24.97	-0.39	24.58	25.36	26.13
	3	21	0.36	19.78	20.64	21.51
		42.99	-0.6	42.93	43.59	44.26
	1	59.44	0.96	57.02	58.48	59.94
B52	5	5.169	0.08	4.908	5.089	5.27
		20.9	-0.07	20.5	20.97	21.45
	3	17.11	0.11	16.56	17	17.44
		41.13	0.41	40.22	40.72	41.22
	1	55.14	-0.44	54.27	55.58	56.9
F16	5	18.91	0.16	18.21	18.75	19.28
		37.37	-0.25	36.33	37.62	38.9
	3	36.89	-0.21	35.55	37.1	38.66
		43.64	-0.6	43.14	44.24	45.35
	1	71.6	0.82	69.38	70.78	72.17

All sixty of the research model results lie within the simulation 95% confidence intervals. Below is a histogram plot of these differences.

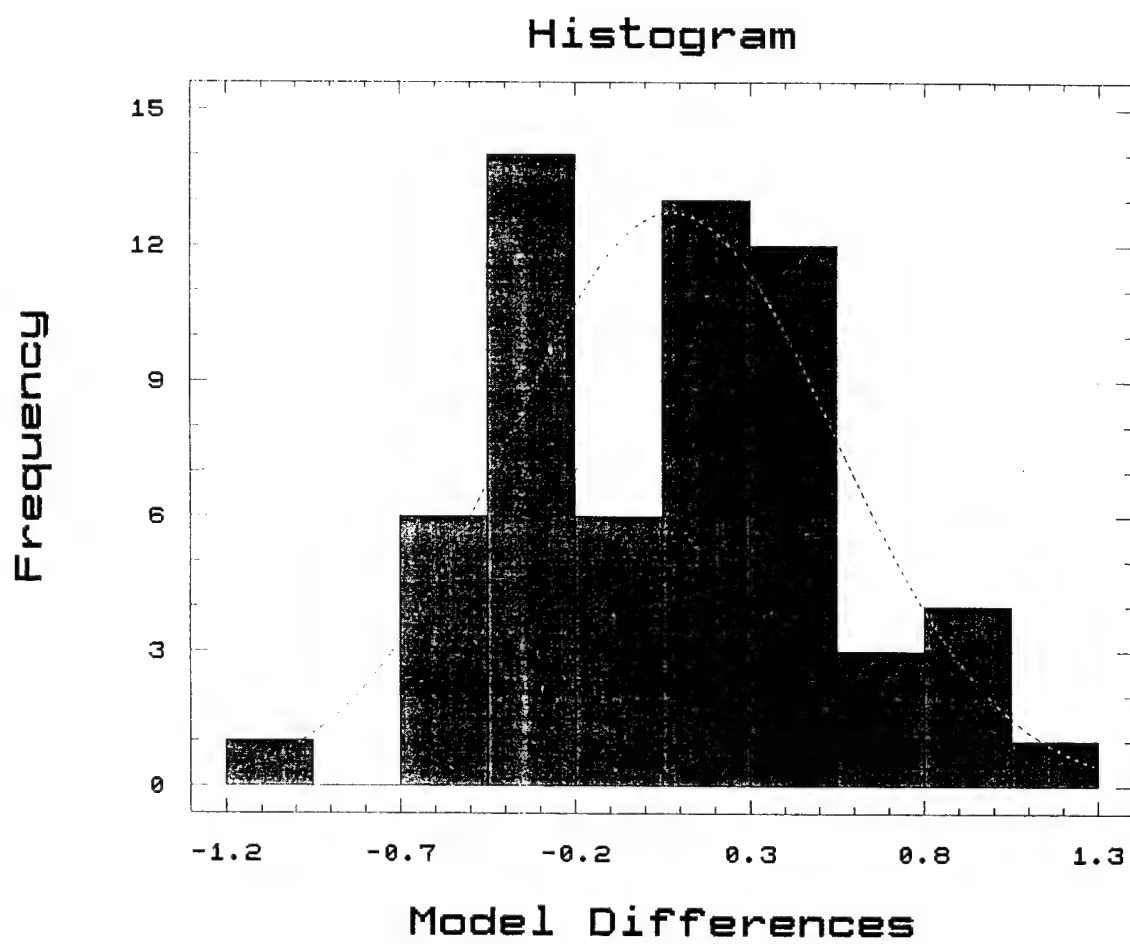


Figure 55: Histogram of component redundancy model differences

An analysis of the model differences provides the following statistics output from Execustat (Strategy Plus, 1993):

```
-----  
Sample size = 60  
Mean = 0.0783833  
Variance = 0.221055  
Std. deviation = 0.470165  
  
95% confidence intervals  
Mean: (-0.0430734,0.19984)  
Variance: (0.158818,0.328902)  
Std. deviation: (0.39852,0.5735)  
-----
```

Figure 56: Sample mean and variance of differences -
component redundancy case

The confidence intervals and the hypothesis means test assume that the sample is from a normal distribution. A normal probability plot of the differences is given below:

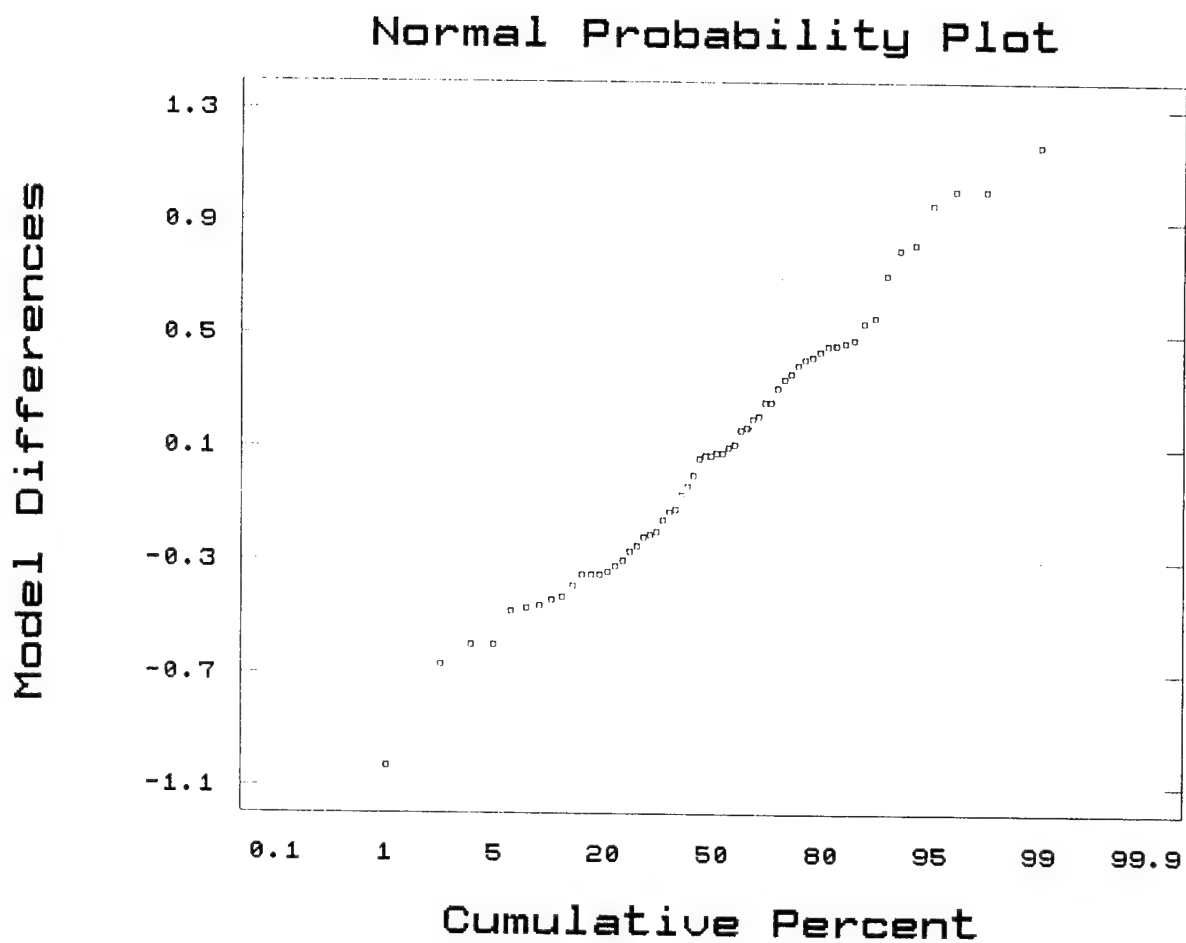


Figure 57: Normal probability plot - redundancy case

To test the hypothesis that the sample is from a normal distribution, a Shapiro-Wilks test was used. This test was discussed in section 4.2.1. Using Execustat, the test statistic computed by the program was 0.977731 and the P-value was 0.5830. This implies that at an $\alpha = 0.05$, the null hypothesis that the sample is from a normal distribution cannot be rejected.

Now the test of hypothesis on the model differences can be performed. The test is shown below:

Let $D_i = (\text{research output}_i - \text{simulation output}_i)$ and \bar{D} and S^2 be the sample mean and variance, then the two-sided test is:

$$H_0: \bar{D} = 0$$

$$H_a: \bar{D} \neq 0$$

with the following test statistic:

$$t_0 = \frac{\bar{D} - 0}{0.470165/\sqrt{60}}$$

This test statistic follows a t distribution with 59 degrees of freedom if H_0 is true. After t_0 is calculated, H_0 is rejected if either $t_0 > 2.00$ or $t_0 < -2.00$ where 2.00 and -2.00 are the upper and lower 0.025 percentage points

of the t distribution with 59 degrees of freedom. Execustat gave the following t -statistic of 1.29137 and P -value of 0.2016. Based on an α value of 0.05, the null hypothesis that the model differences are equal to zero cannot be rejected. The power of the test is very high (β very small) using a difference value of 2.5%. The c -value is 5.317 which is off the OC charts in Hines and Montgomery (1980). The research model appears to perform well in the component redundancy case. The next section considers response timeliness of the research method.

4.3 Timing Issues

As discussed in section 2.4 and section 3.4, the ability to obtain results in a timely manner is very important to the practitioners. The following is a brief comparison of the run times between the research model and the simulation model. Because of the large number of runs and the time necessary to individually time each run, not every run made in this chapter was timed. The following runs were considered as representative of the worst and best possible performers for each model type.

Table 34: Timing Study Scenarios

Model	Worst Time	Best Time
Simulation	1. F-16C, 5 A/C, Redundancy, Full Stock	2. F-16C, 1 A/C, No Redundancy, Zero Stock
Research	3. F-15C, 1 A/C, No Redundancy, Zero Stock	4. F-16C, 3 A/C, Redundancy, Full Stock

The table below lists the results of timing each of these scenarios for both models. The results are all in minutes and the scenario numbers correspond to the numbers given in the table above.

Table 35 : Timing Comparison of Research versus Simulation

Model

Scenario	Research Run Time	Simulation Run Time
1	< 0:01	4:42
2	< 0:01	1:52
3	< 0:01	1:57
4	< 0:01	3:21

The runs were made on a Dell Low Profile Pentium ISA/PSI system with a Intel Comparative Microprocessor Performance (iComp) index rating of 735/90 and an internal clock speed of 90 Mhz. As the reader can see the research model is

significantly faster than the simulation model at obtaining results. This would be especially critical if multiple runs were necessary for a sensitivity analysis. Another issue is the fact that the simulation run times are sensitive to the number of aircraft and number of parts being analyzed. Since the simulation model could only handle 40 part-types, the impact of large numbers of parts in the system could not be tested adequately but increasing the number of aircraft definitely increased run times. The research model on the other hand saw little impact from the different scenarios. The biggest drivers in run times for this model will be number of components and the number of cycles through the code. But even in scenario three above which cycled through the code 16 times, the run time was minimal. To further illustrate the power of this method, another run of the research model was made using scenario three above but placing 100 parts in the system. The run time was still a short 1.5 seconds.

4.4 Optimization Tool Implementation

The following is a description of the input file and model changes necessary to implement the optimization

method described in section 3.5. There are also several examples of how this method might be useful.

The additional input file necessary to implement this new method is simple and consists of only three fields. The first is the availability goal desired. The second field defines the step size or change in operating hours per system per day between each cycle. The third is simply a numeric input field that the user can use to identify different output files. This field is simply read into the program and then wrote out to the output file as a way of identifying the file. If the user was only interested in the current availability of the entire system, he could use an availability goal of 100%. As one can see from the stopping rules given above, this would result in one pass through the system and a report of the current availability.

The additional code necessary to implement this optimization is shown in the figure below:

```

Program Main

CHARACTER*13 LIMITNSN(5),hLN(5)
INTEGER AC,pass2,upac
REAL*8 avail1,avail3,avail2,LIMITEOJ(5),stklvl,goal
1,havail2,havail1,hEOJ(5),flyhrs,sorties, havail3
real*4 step

OPEN(UNIT=9,FILE='DISC1.INP',STATUS='OLD')
open(unit=15,file='comb2.out',status='old')

```

```

read(9,*) goal,step,stk1v1

pass2=0

23  call comp(pass2,step,avail1,avail2,avail3,limitnsn
    1,limiteoj,flyhrs,sorties,ac)
    upac=ac-1

    if (ac.le.2.and.avail1.gt.goal) then
        havail1=avail1
        havail2=avail2
        havail3=avail3
        do 420 i=1,5
            hln(i)=limitnsn(i)
            heoj(i)=limiteoj(i)
420    continue
        pass2=pass2+1
C      print *,flyhrs,sorties
C      print *,stk1v1,flyhrs*sorties,avail1,goal,pass2
C      pause
        write(15,425) stk1v1,flyhrs*sorties,avail1
        goto 23
    endif

    if (avail2.ge.goal.and.ac.gt.2) then
        havail1=avail1
        havail2=avail2
        havail3=avail3
        do 400 i=1,5
            hln(i)=limitnsn(i)
            heoj(i)=limiteoj(i)
400    continue
        pass2=pass2+1
C      print *,flyhrs,sorties
C      print *,stk1v1,flyhrs*sorties,avail2,goal,pass2
C      pause
        write(15,425) stk1v1,flyhrs*sorties,avail2
        goto 23
    endif

425  format(1x,f8.3,f8.3,f8.3)
    if (pass2.gt.0) then
C      print *,flyhrs,sorties
        flyhrs=flyhrs - (step/sorties)
        flyhrs=dint(flyhrs*1e5+0.1)/1e5
C      print *,flyhrs,sorties
        avail1=havail1
        avail2=havail2
        avail3=havail3
        do 410 i=1,5
            limitnsn(i)=hln(i)
            limiteoj(i)=heoj(i)
410    continue
    endif
$page

```

```

        print *, ' '
        print *, ' '
        print 628, sorties*flyhrs
628    format(1x, 'The total flying hours per a/c per day is '
1,F7.3)
        print *, ' '
        print 630, ac, avail1
        print 640, upac, avail2
        print 650, upac, avail3
630    FORMAT(1X, 'The availability of', I3, ' systems is
', 2P, E18.11)
640    FORMAT(1X, 'The availability of', I3, ' or more systems is '
1, 2P, E18.11)
650    FORMAT(1X, 'The availability of', I3, ' systems is
', 2P, E18.11)
        PRINT *, ' '
        PRINT *, ' This is a list of the top 5 problem parts'
        PRINT *, ' Stock Number           E[0]'

        DO 655 J=5, 1, -1
            IF (LIMITNSN(J).GT. '          ') THEN
                PRINT 656, LIMITNSN(J), LIMITEOJ(J)
            ENDIF
656    FORMAT(1X, A13, 5X, 1P, E9.3)
655    CONTINUE
        end

```

Figure 58: Code to implement flying hour optimization

The code written for the earlier method simply becomes a subroutine call in this program with minor changes in the original to update the operating hour program and avoid unnecessary recalculation of binomial and hypergeometric distributions. On the last page of appendix A, the additional lines of code necessary to implement this change in the original code are given in the context of the previously presented code and have been underlined to highlight them.

The following example illustrates how the capability in section 3.3 might be used. In section 4.2.4.1, the B-52 three aircraft availability of two or more aircraft was 90.01%. What if the availability goal was only 80%? How many additional flying hours per aircraft per day could someone achieve and still meet the availability goal? If this information is used in the model, we discover that instead of 7.2 flying hours per aircraft, we can get 10.20 flying hours per aircraft while still achieving a 81.44% availability. This process could also be used to study the relationship between different stock levels and flying hours and their impact on aircraft availability. The plot given below shows this relationship for the B-52 three aircraft case discussed above.

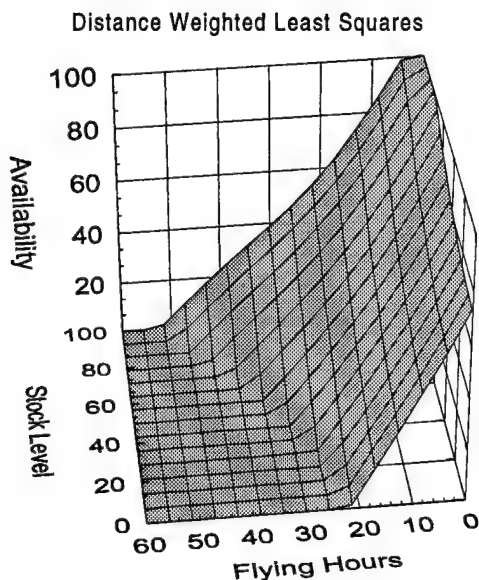


Figure 59. Relationship of flying hours and stock level to availability

This graph was generated by simply running the model with 1% availability goals and a step size of 0.5 flying hours. The stock levels were decremented from 100% by randomly selecting the missing items for each additional loss in available stock until zero was reached.

This graph allows the general assessment of any combination of flying hour and stock level decisions, although the specific stock level combinations may result in somewhat varied availabilities.

CHAPTER 5

SUMMARY AND FUTURE WORK

5.1 Summary

Logistics support has always been a key element of combat effectiveness as well as an important element in commercial transportation systems. The Air Force is interested in developing tools and methods to assess the impact of logistics on combat capability. The current method used by the Air Force to assess aircraft spare's support was developed for use with a large number of aircraft. With the current military drawdown and the high cost of new aircraft, fewer aircraft are fielded and therefore, some of the critical assumptions made in the current model are unsatisfactory. This includes such things as the assumption of an infinite calling population of demands and the distribution of backorders by sampling with replacement. The problem is well-documented (Faughaber, 1993) and can occur with as many as eight aircraft (Miyares, 1995). This research develops an original method to assess the system availability of a small number of vehicles or machines which overcomes some

of the flaws of existing models. This small calling population of machines is also known as a finite-source queueing environment. The research method also incorporates backorder distributions based on sampling without replacement (Hypergeometric) which is more accurate in this environment than the more common method of sampling with replacement (Binomial). Although the case study for this work is a military application, it is applicable to any small system of vehicles or machines where demand and repairs follow an exponential distribution. The new method also allows the user to include component redundancy and component spares in the system. The basic research method is summarized in the figure below:

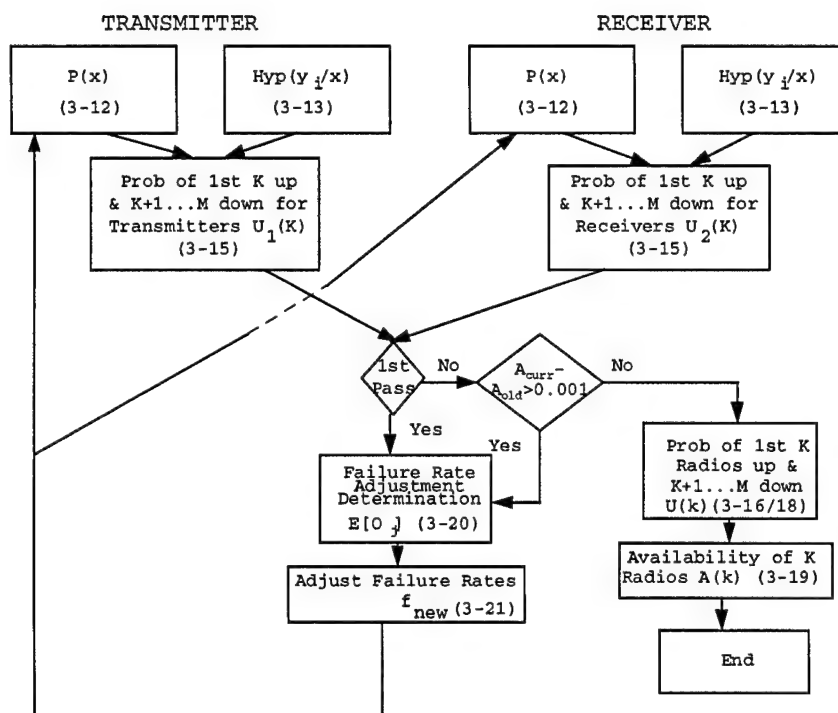


Figure 60. Summary of research method

Although this general approach can be computationally burdensome, this work proposes two original methods to significantly reduce the number of computations necessary to find a solution. They include eliminating the tails of individual failure distributions and, selectively, excludes other components because of their insignificant downtime contributions to the system. The first method involves "batching" the components in groups to reduce the number of computations. The research presents a method to solve this NLIP problem which minimizes the number of computations

necessary. The second method of eliminating parts uses the initial values of $U_j(k)$'s to rank order and prioritize the components that are ignored until a user-defined maximum accuracy loss is reached. The research also shows that this loss is an upper limit and the loss will usually be much smaller.

In order to test the research method, a FORTRAN program was written to implement the approach. Using actual U.S. Air Force failure, repair, and scenario data for the F-16, F-15, E-3B, and B-52H, the results were compared to a simulation which provided "live" results. These aircraft were selected because they represent a good cross-section of the Air Force inventory. The number of aircraft and stock levels used in the tests were representative of Air Force usage and were approved by the Air Force. They also provided a broad range of conditions from which to exercise the model. The research model was tested under two different cases: component redundancy and no component redundancy. The results are detailed in Chapter Four and, in both cases, the research model performed very closely to the simulated "live" data. In only three cases of 120 were the research results outside of the 95% confidence intervals for the simulation model and the largest 95% simulation confidence interval on

availability was less than 4% wide. In addition, both hypothesis of means tests on the two different case output differences had high power values. This provides strong evidence that the research method performs well under a wide range of operating conditions. The research model has several advantages over the simulation model which include speed of processing and the ability to handle more components in the system.

Finally, an additional capability was added to the research method that doesn't exist in the current model or in the simulation model. An optimization technique is provided that allows the user to optimize operating hours given a system availability goal or target. This new ability allows the user to estimate the additional capability available (i.e. additional operating hours) if he is currently exceeding his availability goal. The usefulness of this capability was demonstrated in Chapter Four. The next section discusses areas available for future work.

5.2 Future Work

5.2.1 Research Model Enhancement

There are several possible avenues to pursue that would enhance the usefulness of the research method. First, the current research model assumes that there is no cannibalization of components between vehicles or systems. In some situations, this could be a pessimistic assumption and methods to allow partial or full cannibalization of components should be explored. This researcher believes that modifying eq. 3-15 shown below would be a good place to start in implementing this feature.

$$U_j(k) = \sum_{x=0}^{\infty} P_j\{X = x\} \sum_T \text{hyp}(y_1, y_2, \dots, y_M | x) \quad I \leq k \leq M \quad (3-15)$$

By modifying the set T used in the hypergeometric calculations, it may be possible to account for cannibalization. In addition, it may also be necessary to modify eq. 3-16, 17, and 18 to account for the affect of cannibalization.

Another aspect that should be explored is adding sub-components in the computations. This effort could take a

number of routes. First, it may be possible to simply extend the current method by adding "duplicate" layers of computations where each level of component and sub-component feeds the next higher assembly. One potential problem with this approach is the continued possibility of computational explosion. Another possible solution that shows promise is the use of a weighting factor that would adjust the component's repair rate based on the availability of sub-components. Since subcomponents don't have a direct impact on the system availability, this appears to be a viable alternative and is less likely to carry the high computational load that the first approach does.

Currently, the model only allows fixed demand or repair rates. Allowing time-varying failure or repair rates would also be a definite enhancement. This change should primarily impact the form of eq. 3-12 shown below which would now need to include a time-based function.

$$P(X = x) = \begin{cases} P(0) \frac{f^x R^x}{x!}, & x \leq s \\ P(0) \frac{f^x R^s}{x!} \frac{R!}{(R - x + s)!}, & s + 1 \leq x \leq R + s \end{cases} \quad (3-12)$$

Accuracy and computational loads are issues here too.

Finally, one implementation issue in the current research model should be addressed. It is the ability of the FORTRAN code to only process components with QPA's of three or less. This should be changed by expanding the code that implements eq. 3-15 mentioned above.

5.2.2 B-1 Case Study

HQ ACC conducted an actual deployment test of B-1 aircraft to Roswell, New Mexico in November 1994. The fortunate timing of the B-1 test in Roswell provided a unique opportunity to acquire live data. The data collected from this test could form the basis of a live data set for use in testing the cannibalization enhancement mentioned in section 5.2.1. To ensure a reliable comparison between the actual aircraft performance and the new method's predicted performance, the enhanced research method will use the same scenario and assumptions that were used during the B-1 test. With this input data, the model and the actual results could be compared.

The B-1 test would satisfy the inherent assumptions necessary for use in this research based on the B-1 test plan (HQ AFOTEC, 1994) and based on experience from the Coronet Warrior exercises (Lewis, 1987). One of the

primary purposes of the test was to study the stand-alone deployment capability of the B-1 so outside resupply was be restricted. Capt Sullivan (1994) from HQ ACC and Lt. Col Pipp (1994) from HQ AF/LEYS also agree the test satisfies the scope and limitations of this research. Capt Sullivan's branch planned and managed the test for HQ ACC.

The actual exercise lasted 14 days with 9 aircraft. Since all the aircraft were operational when the exercise began, the first six days were used to "steady-state" the system and fill the repair pipelines. Then the aircraft were operated for eight additional days. The average availability on each of the eight days was eight. They had demands for 58 components during the exercise and the actual demand rates for these parts would be used to build the model input files. Because of the small number of parts sent through the repair pipeline, one repair time average was computed for all parts and would be used as a model input. Although the unit size is somewhat large for this research's scope, this data could provide live data for validating an enhanced model.

REFERENCES

- Bachmann, A. 1986. Implementing A Periodic Review (r,q) Model with Constraints. *Inventory in Theory and Practice*. Elsevier Publishing Co., Inc., New York.
- Baccelli, F., and Bremaud, P. *Elements of Queueing Theory - Palm-Martingale Calculus and Stochastic Recurrences*. Springer-Verlag, Berlin.
- Balsamo, S., Clo, M., and Donatiello, L. 1992. Cycle Time Distribution of Cyclic Networks with Blocking. *Queueing Networks With Finite Capacity*. North-Holland, New York.
- Banks, J and Carson II, J. 1984. *Discrete-Event System Simulation*. Prentice Hall, Inc., Englewood Cliffs.
- Banks, J., and Fabrycky, W. 1987. *Procurement And Inventory Systems Analysis*. Prentice Hall, Inc., Englewood Cliffs.
- Beyer, W., 1986. *CRC Handbook of tables for Probability and Statistics, 2nd Edition*. CRC Press, Inc., Boca Raton.
- Bhat, U., and Basawa, I. 1992. *Queueing and Related Models*. Oxford University Press, Oxford.
- Bittel, L. 1991. *Right on Time!: The Complete Guide for Time-Pressured Managers*. McGraw-Hill, Inc., New York.
- Blackburn, J. 1991a. Just-In-Time: The Genesis Of Time Compression. *Time-Based Competition: The Next Battleground In American Manufacturing*. Business One Irwin, Homewood.
- Blackburn, J. 1991b. The Time Factor. *Time-Based Competition: The Next Battleground In American Manufacturing*. Business One Irwin, Homewood.
- Blanchard, B. 1986. *Logistics Engineering and Management*. Prentice Hall, Inc., Englewood Cliffs.

REFERENCES

- Bachmann, A. 1986. Implementing A Periodic Review (r,q) Model with Constraints. *Inventory in Theory and Practice*. Elsevier Publishing Co., Inc., New York.
- Baccelli, F., and Bremaud, P. *Elements of Queueing Theory - Palm-Martingale Calculus and Stochastic Recurrences*. Springer-Verlag, Berlin.
- Balsamo, S., Clo, M., and Donatiello, L. 1992. Cycle Time Distribution of Cyclic Networks with Blocking. *Queueing Networks With Finite Capacity*. North-Holland, New York.
- Banks, J and Carson II, J. 1984. *Discrete-Event System Simulation*. Prentice Hall, Inc., Englewood Cliffs.
- Banks, J., and Fabrycky, W. 1987. *Procurement And Inventory Systems Analysis*. Prentice Hall, Inc., Englewood Cliffs.
- Beyer, W., 1986. *CRC Handbook of tables for Probability and Statistics, 2nd Edition*. CRC Press, Inc., Boca Raton.
- Bhat, U., and Basawa, I. 1992. *Queueing and Related Models*. Oxford University Press, Oxford.
- Bittel, L. 1991. *Right on Time!: The Complete Guide for Time-Pressured Managers*. McGraw-Hill, Inc., New York.
- Blackburn, J. 1991a. Just-In-Time: The Genesis Of Time Compression. *Time-Based Competition: The Next Battleground In American Manufacturing*. Business One Irwin, Homewood.
- Blackburn, J. 1991b. The Time Factor. *Time-Based Competition: The Next Battleground In American Manufacturing*. Business One Irwin, Homewood.
- Blanchard, B. 1986. *Logistics Engineering and Management*. Prentice Hall, Inc., Englewood Cliffs.

- Blanchard, B., and Fabrycky, W. 1990. *Systems Engineering and Analysis*. Prentice Hall, Inc., Englewood Cliffs.
- Borland, 1991. *dBase III Plus*. Borland International, Inc., Scotts Valley.
- Brown, R. 1987. *Production and Inventory Control Handbook*. McGraw-Hill Book Co., New York.
- Buffa, E., and Taubert, W. 1972. *Production-Inventory Systems: Planning and Control*. Richard D. Irwin, Inc., Homewood.
- Bunday, B., and Scranton, R. 1980. The G/M/r Machine Interference Model. *European Journal of Operational Research* **4**, 399-402.
- Buzen, J. 1973. Computational Algorithms for Closed Queueing Networks with Exponential Servers. *Communications of the ACM* **16**, 527-531.
- Chakravarthy, S. 1992. A Finite Capacity Queueing Network with Single and Multiple Processing Nodes. *Queueing Networks With Finite Capacity*. North-Holland, New York.
- Clark, A. 1958. A Dynamic, Single Item, Multi-Echelon Inventory Model. The Rand Corp., RM2297.
- Coll, R., Coll, J., and Rein, D. 1991. The effect of computerized decision aids on decision time and decision quality. *Information & Management* **20**, 75-81.
- Conover, W. 1980. *Practical Nonparametric Statistics*, 2nd ed.. John Wiley & Sons, New York.
- Dalkey, N. 1968. *Simulation. Systems analysis and Policy Planning: Applications in Defense*. American Elsevier Publishing Co., New York.
- Daniel, H. 1948. *For Want Of A Nail: The Influence of Logistics on War*. Whittlesay House, New York.
- Demmy, W., and Hobbs, J. 1983. Dyna-METRIC: An Overview. *Air Force Journal of Logistics*, Spring, 14-17.
- DRC. 1993a. *Dyna-METRIC Microcomputer Analysis System*. Dynamics Research Corporation (DRC), Andover.

- . 1993b. *Major Command Dyna-METRIC Microcomputer Analysis System*. Dynamics Research Corporation (DRC), Andover.
- Faughaber, K. 1993. DMAS Small PAA Utility. HQ USAF/LGSS, Washington D.C.
- Feeney, G., and Sherbrooke, C. 1966. The (s-1,s) Inventory Policy Under Compound Poisson Demand. *Management Science* **12**, 391-411.
- Gage, T., and Ogan, A. 1983. Dyna-Metric: A Valuable Management Tool. *Air Force Journal of Logistics*, Spring, 22-24.
- Gelenbe, E., Pujolle, G., and Nelson, J. 1987. *Introduction to Queueing Networks*. John Wiley & Sons, New York.
- Gnedenko, B., and Kovalenko, I. 1989. *Introduction to Queueing Theory*. Birkhauser, Boston.
- Graves, S. 1985. A Multi-Echelon Inventory Model for a Repairable Item with One-for-One Replenishment. *Management Science* **31**, 1247-1256.
- Greene, J. 1974. *Production and Inventory Control: Systems and Decisions*. Richard D. Irwin, Inc., Homewood.
- Gross, D., and Harris, C. 1985. *Fundamentals of Queueing Theory*. John Wiley & Sons, Inc., New York.
- Gross, D., Miller, D., and Soland, R. 1983. A Closed Queueing Network Model for Multi-Echelon Repairable Item Provisioning. *IIE Transactions* **15**, 344-352.
- Gross, D., Soland, R., and Pinkus, C. 1981. Designing a Multi-Product, Multi-Echelon Inventory System. *Multi-Level Production/Inventory Control Systems: Theory and Practice*. North-Holland, New York.
- Hadley, G., and Whitin, T. 1963. *Analysis of Inventory Systems*, pg. 204-212, Prentice-Hall, Englewood Cliff.
- Hale, Capt. J., 1995. Approval of Small PAA Model Concept - Memorandum. HQ ACC/LGSIP, Langley AFB, VA., 14 Jul 95.

- Hillestad, R., and Carrillo, M. 1980. Models and Techniques for recoverable item stockage when demand and the repair process are non-stationary--Part I: Performance Measurement. The RAND Corp., N-1482-AF.
- Hillestad, R. 1982. Dyna-METRIC: Dynamic Multi-Echelon Technique for Recoverable Item Control. The RAND Corp., R-2785-AF.
- Hillier, F., and Lieberman, G. 1986. *Introduction to Operations Research*. Holden-Day, Inc., Oakland.
- Hines, W., and Montgomery, D. 1980. *Probability and Statistics in Engineering and Management Science*. John Wiley & Sons, New York.
- HQ AFOTEC, 1994. The B-1B Operational Readiness Assessment. Kirtland AFB, New Mexico, 31 January 1994.
- Isaacson, K., and Boren, P. 1988. Dyna-METRIC Version 5: A Capability Assessment Model Including Constrained Repair and Management Adaptations. The RAND Corp., R-3612-AF.
- Isaacson, K., and Boren, P. 1993. Dyna-METRIC Version 6 An Advanced Capability Assessment Model. The Rand Corp., R-4214-AF.
- Isaacson, K., Boren, P., Tsai, C., and Pyles, R. 1988. Dyna-METRIC Version 4: Modeling Worldwide Logistics Support of Aircraft Components. The RAND Corp., R-3389-AF.
- Jackson, J. 1957. Networks of Waiting Lines. *Operations Research* **5**, 518-521.
- , 1963. Jobshop-like Queueing Systems. *Management Science* **10**, 131-142.
- Janson, R. 1987. *Statistical Measures for Production and Inventory Management*. Prentice Hall, Inc., Englewood Cliffs.
- Johnson, L., and Montgomery, D. 1974. *Operations Research in Production Planning, Scheduling, and Inventory Control*. John Wiley & Sons, Inc., New York.

- Jou, Y., Nilsson, A., and Lai, F. 1992. The Upper Bounds for Performance of a Finite Capacity Polling System under Bursty Arrivals. *Queueing Networks With Finite Capacity*. North-Holland, New York.
- Kalashnikov, V. 1994. *Mathematical Methods in Queuing Theory*. Kluwer Academic Publishers, Boston.
- Kaplan, A. 1989. Incorporating Redundancy Considerations into Stockage Policy Models. *Naval Research Logistics* **36**, 625-638.
- Khoshnevis, B. 1994. *Discrete Systems Simulation*. McGraw-Hill, Inc., New York.
- King, Dr. R., 1993. Analyst, Telephone conversation. Logistics Management Institute, Alexandria, Va., 8 June 1993.
- Law, A., and Kelton, W. 1991. *Simulation Modeling & Analysis*. McGraw-Hill, Inc., New York.
- Lawler, E., and Bell, M. 1966. A Method for Solving Discrete Optimization Problems. *Operations Research* **14**, 1098-1112.
- Lee, D-S, and Sengupta, B. 1992. Two Queues in Tandem with Dropping or Blocking of Customers. *Queueing Networks With Finite Capacity*. North-Holland, New York.
- Lee, S., Moore, L., and Taylor, B. 1990. *Management Science*. Allyn and Bacon, Boston.
- Lee, H., and Nahmias, S. 1993. Single Product, Single Location Models. *Logistics of Production and Inventory*. North-Holland, New York.
- Lewis, C. 1970. *Scientific Inventory Control*. American Elsevier Publishing Co., Inc., New York.
- Lewis, T. 1987. *Unit Level WRSK Assessment and Sortie Generation Simulation Model*. MS Thesis, Air Force Institute of Technology, Wright Patterson AFB, OH.
- Love, S. 1979. *Inventory Control*. McGraw-Hill Book Co., New York.

- Mace, A. 1964. *Sample-Size Determination*. Reinhold Publishing Corp, New York.
- Medhi, J. 1991. *Stochastic Models in Queueing Theory*. Academic Press, Inc., New York.
- Mendenhall, W., Scheaffer, R., and Wackerly, D. 1986. *Mathematical Statistics with Applications*. Duxbury Press, Boston.
- Meyer, C. 1993. The Character of a Finite Markov Chain. *Linear Algebra, Markov Chains, and Queueing Models*. Springer-Verlag, New York.
- Microsoft, 1993. *FORTTRAN Version 5.1*. Microsoft Corp., Redmond.
- Mirasol, N. 1964. A Queueing Approach to Logistics Systems. *Operations Research* **12**, 707-724.
- Miyares, Capt. P., 1993. Chief, Weapon System Assessment Branch. Telephone conversation. HQ ACC/LGSW, Langley AFB, VA., 8 June 1993.
- Miyares, Capt. P., 1995. Validation of Aircraft Combat Capability Model - Memorandum. HQ ACC/LGSW, Langley AFB, VA., 24 Mar 95.
- Montgomery, D. 1991. *Design and Analysis of Experiments*, 3rd ed. John Wiley & Sons, New York.
- Moore, F., and Hendrick, T. 1977. *Production/Operations Management*. Richard D. Irwin, Inc. Homewood.
- Muckstadt, J. 1973. A Model For A Multi-Item, Multi-Echelon, Multi-Indenture Inventory System. *Management Science* **20**, 472-481.
- , 1976. The Consolidated Support Model (CSM): A Three-Echelon, Multi-Item Model for Recoverable Items. The RAND Corp., R-1923-PR.
- Nahmias, S. 1982. Perishable Inventory Theory: A Review. *Operations Research* **30**, 680-708.

- O'Malley, T. 1983. *The Aircraft Availability Model: Conceptual Framework and Mathematics*. Logistics Management Institute, Washington, D.C.
- Palm, C. 1938. Analysis of the Erlang Traffic Formulae for Busy-Signal Arrangements. *Ericsson Technics*. 4, 39-58.
- Pegden, C. 1990. *Introduction to Simulation Using SIMAN*. McGraw-Hill, Inc., New York.
- Perros, H. 1994. *Queueing Networks with Blocking*. Oxford University Press, Oxford.
- Pipp, Lt. Col. D., 1994. Supply Policy analyst. Telephone Conversation. HQ AF/LEYS, Alexandria, VA., 23 Mar 1994.
- Plossl, G. 1985. *Production and Inventory Control: Principles and Techniques*. Prentice Hall, Inc., Englewood Cliffs.
- Pritsker, A. 1986. *Introduction to Simulation and SLAM II*. John Wiley & Sons, New York.
- Pyles, R. 1984. The Dyna-METRIC Readiness Assessment Model: Motivation, Capabilities, and Use. The RAND Corp., R-2886-AF.
- Pyles, R., and Tripp, R. 1983. Dyna-Metric: A Valuable Management Tool. *Air Force Journal of Logistics*, Spring, 18-21, 24.
- Quade, E. 1968. Pitfalls And Limitations. *Systems analysis and Policy Planning: Applications in Defense*. American Elsevier Publishing Co., New York.
- Rivett, P. 1980. *Model Building for Decision Analysis*. John Wiley & Sons, New York.
- Rouse, W. 1986. Design and Evaluation of Computer-Based Decision Support Systems. *Microcomputer Decision Support Systems: Design, Implementation, and Evaluation*. QED Information Sciences, Inc., Wellesley.
- Sage, A. 1986. An overview of Contemporary Issues in the Design and Development of Microcomputer DSS. *Microcomputer Decision Support Systems: Design,*

Implementation, and Evaluation. QED Information Sciences, Inc., Wellesley.

Sankar, Y. 1991. *Management Of Technological Change*. John Wiley & Sons, New York.

Sebring, Smsgt. C. (Ret.), 1994. Supply Analyst. Telephone conversation. 27 May 1994.

Shannon, R. 1975. *System Simulation: the art and science*. Prentice-Hall, Inc., Englewood Cliffs.

Sharma, O. 1990. *Markovian Queues*. Ellis Horwood, New York.

Sherbrooke, C. 1966. METRIC: A Multi-Echelon Technique for Recoverable Item Control. The RAND Corp., RM-5078-PR.

-----, 1968a. METRIC: A Multi-Echelon Technique For Recoverable Item Control. *Operations Research* **16**, 122-141.

-----, 1968b. A Management Perspective on METRIC-Multi-Echelon Technique for Recoverable Item Control. The RAND Corp., RM-5078-PR.

-----, 1971. An Evaluator For The Number Of Operationally Ready Aircraft In A Multilevel Supply System. *Operations Research* **19**, 619-635.

-----, 1986. Vari-METRIC: Improved Approximations For Multi-Indenture, Multi-Echelon Availability Models. *Operations Research* **34**, 311-319.

-----, 1992. *Optimal Inventory Modeling of Systems: Multi-Echelon Techniques*. John Wiley & Sons, Inc., New York.

Slay, M. 1980. VARI-METRIC- An Approach to Modeling Multi-Echelon Resupply when the Demand Process is Poisson with a Gamma Prior. Working Paper. Logistics Management Institute, Washington D.C.

Slay, M., and King, R. 1987. Prototype Aircraft Sustainability Model. Logistics Management Institute, Washington D.C., Report # AF601-R2.

- Sluis, E. 1993. *Heuristics for Complex Inventory Systems: Deterministic and stochastic problems*. Thesis Publishers, Amsterdam.
- Specht, R. 1968. *The Nature Of Models. Systems Analysis and Policy Planning: Applications in Defense*. American Elsevier Publishing Co., New York.
- Sprague, R., and Carlson, E. 1982. *Building Effective Decision Support Systems*. Prentice-Hall, Inc., Englewood Cliffs.
- Strategy Plus, 1993. *Student Edition of Execustat Version 3.0*. Duxbury Press, Belmont.
- Sullivan, Capt. J., 1994. Headquarters Supply analyst, Telephone conversation. HQ ACC/LGSP, Langley AFB, VA., 4 Mar 1994.
- Takagi, H. 1993. *Queueing Analysis - A Foundation of Performance Evaluation - Volume 2: Finite Systems*. North-Holland, New York.
- Thierauf, R. 1975. *Systems Analysis and Design of Real-Time Management Information Systems*. Prentice-Hall, Inc., Englewood Cliffs.
- Thierauf, R. 1988. *User-Oriented Decision Support Systems: Accent on Problem Finding*. Prentice-Hall, Inc., Englewood Cliffs.
- Tripp, R., Cohen, I., Pyles, R., Hillestad, R., Clarke, R., Limbert, S., and Kassicieh, S. 1991. A Decision Support System for Assessing and Controlling the Effectiveness of Multi-Echelon Logistics Actions. *Interfaces* **21**, 11-25.
- Tsai, C. 1989. Dyna-SCORE: Dynamic Simulation of Constrained Repair. The RAND Corp., R-3637-AF.
- U.S. Dept of the Air Force, 1990. WRM Program. AFR 400-24, Washington D.C., 31 Jul 1990.
- U.S. Dept. of the Air Force, 1993. U.S. Air Force Supply Manual. AFM 67-1 Vol II Part 2, Washington D.C., 1 Oct 1993.

Waters, C. 1992. *Inventory Control And Management*. John Wiley & Sons, Inc., New York.

Weapon System Assessment Branch, 1994. *DMAS Training Material*. Air Combat Command Headquarters, Langley Air Force Base, Virginia.

APPENDIX A
RESEARCH MODEL CODE

```

      PROGRAM MAIN
C      COMMON/MINE/REPAIR(40),NRTS(40),QPA(40),MINQPA(40),run,STK(30)
C      1,SQPA(30),TLRU(40),SRULOC(40),SRUDM(30),MSOR(30),FMC(30),F(40)
C      2,FRT(40),LSTK(40)
C--- only need above structure if using SRUs

      CHARACTER*13 LIMITNSN(5),NSN(100),NUM
      INTEGER X,Q,STKS,LRUAV,NOFAIL,R,AC,P,BOFL,SRU,QPAS,MINQ
      1,PASS,WORST(30),lrupass,upac,lrucnt,trun,batch,SRUAV,RORST
      REAL*8 FIX,FAIL,FLYHRS,BORDER(100,0:40),HYPERT1(10,10,0:10,0:40)

      1,U(100,0:10),v(100,0:10),UTOT(0:10),BO,WORSTEOJ(30),EOJ,PO,avail1
      2,HYPER(100,0:100),FRT(100),REPAIR(100),avail3,avail2,LIMITEOJ(5)
      3,RORSTEOJ,CHANGE,WORSTE
      real*4 DELTA,NRTS(100),QPA(100),MINQPA(100),LSTK(100)
      1,tlru(100)

      OPEN(UNIT=7,FILE='DISC.INP',STATUS='OLD')
      OPEN(UNIT=8,FILE='SRU.INP',STATUS='OLD')
      OPEN(UNIT=17,FILE='comb.out',STATUS='OLD')

C
C      READS IN INITIAL VALUES AND FAILURE RATES
C
C      AC - IS THE NUMBER OF AIRCRAFT/VEHICLES USED
C      MINAC - IS THE MINIMUM AIRCRAFT/VEHICLES NEEDED
C      LRU - IS THE NUMBER OF LRUs/COMPONENTS IN THE DATA SET
C      SORTIES - THIS THE NUMBER SORTIES/TRIPS PER DAY
C      FLYHRS - THIS IS THE NUMBER OF OPERATING HOURS PER SORTIE
C      CANN - THIS TELLS WHETHER PARTS CAN BE CANNED OR NOT
C
C      comment      CALL TIMER(DELTA)
      read(7,*) trun
      read(7,*) batch
      ISRU=0
      AM=0

      READ(7,*) AC
      READ(7,*) MINAC
      READ(7,*) LRU
      READ(7,*) SORTIES
      READ(7,*) FLYHRS
      READ(7,*) CANN
      DO 10 I=1,LRU

C
C      READS IN THE LRU PART NO., DEMAND RATE, QPA, MIN QPA, NO. OF
LRUS
C      IN STOCK, NO. OF SRUS IN LRU, NRTS RATE, REPAIR CYCLE TIME
C
      sru=0
      READ(7,7) NUM,FR,QPAS,MINQ,LRUAV,SRUAV,RTS,REP
      7  FORMAT(A13,F8.5,1X,I3,1X,I3,1X,I3,I3,F5.2,F5.2)
C      PRINT *, NUM,FR,QPAS,MINQ,LRUAV,SRUAV,RTS,REP

```

```

C      PAUSE
      NSN(I)=NUM
      TLRU(I)=SRUAV
C      IF(SRUAV.EQ.0) GO TO 5
C      sru=sru+1
C      ISRU1=ISRU+1
C      ISRU=INT(ISRU+SRUAV)
C      SRULOC(I)=ISRU1
C      TOTAL=0
CC
CC      READS IN SRU DATA
CC      SRU NO., DEMAND RATE, QPA, NO. OF SRUS IN STOCK
CC
C      DO 2 J=ISRU1,ISRU
C      READ(8,*)NU,DM,QPSRU,SRUSTK
C      SQPA(J)=QPSRU
C      STK(J)=SRUSTK
C      TOTAL=TOTAL+DM
C      SRUDM(J)=TOTAL/FR
C 2    CONTINUE
CC
5      NRTS(I)=RTS
      REPAIR(I)=1/REP
      QPA(I)=QPAS
      MINQPA(I)=MINQ
      FRT(I)=FR*FLYHRS*SORTIES
C      PRINT *,FR,FRT(I)
      FRT(I)=DINT(FRT(I)*1E5+0.1)/1E5
C      PRINT *, 'THIS IS FRT',FRT(I), ' AND REPRT ',REPAIR(I)
C      PAUSE
      LSTK(I)=LRUAV
10     CONTINUE

C      END OF DATA INPUT

C      BUILDS HYPER DISTRIBUTION MATRIX

C      R - CREATES THE RANGE FOR HYPER MATRIX

9      R = 30
      CALL HYPERGEO(R,HYPER)
C      DO 17 A=1,R
C      PRINT *,
      HYPER(A,0),HYPER(A,1),HYPER(A,2),HYPER(A,3),HYPER(A,4)
C 17    CONTINUE
C      PAUSE

C
C      THIS CALLS THE SUBROUTINE THAT CALCULATES THE HYPERGEOMETRIC
C      DISTRIBUTION FOR EACH COMBINATION OF MINIMUM QPA, QPA, NUMBER
C      OF UP AIRCRAFT AND THE NUMBER OF COMPONENT BACKORDERS
C
      CALL MYSTUFF(AC,HYPER,HYPERT1)

      WRITE(17,600) AC

```



```

600  FORMAT(1X,'THIS IS THE NUMBER OF AIRCRAFT',I3)
      WRITE(17,610) LRU
610  FORMAT(1X,'THIS IS THE NUMBER OF LRUs ANALYZED',I4)
      WRITE(17,620) SRU
620  FORMAT(1X,'THIS IS THE NUMBER OF SRUs ANALYZED',I4)
      WRITE(17,*) ' '

      print 600, AC
      print 610, LRU
      print 620, SRU
      PRINT *, ' '

C
C      CALCULATES THE FAILURE DISTRIBUTION OF EACH PART - EQ(3-12)
C
C      STKS - STOCK LEVEL
C      FAIL - PART FAILURE RATE
C      FIX - REPAIR RATE
C      Q - R - NUMBER OF COMPONENTS OPERATING
C      X - NUMBER OF COMPONENT FAILURES (IE. DISTRIBUTION #)
C      BO - RESULT OF BACKORDER CALCULATION
C      NOFAIL - MAX POSSIBLE # OF FAILURES - Q+STKS
C      PO - P(0)
C      BORDER(K,J)- # of broken Parts, J, for Part K - Failure
Distribution
C
C
      CHANGE=0
      PASS=1

C      PRINT *, 'THIS IS THE WORSTEOJ INT',WORSTEOJ

24  CONTINUE

      UTOT(AC)=1
      RORST=0
      RORSTEOJ=AC

      DO 25 K=1,LRU

          STKS=INT(LSTK(K))
          IF (PASS.EQ.1) THEN
              FAIL=FRT(K)
          ELSE IF (WORST(I).EQ.K) THEN
              FAIL=FRT(K)
          ELSE
              FAIL=FRT(K) * (WORSTE/AC)
              FRT(K)=FRT(K) * (WORSTE/AC)
          ENDIF
C      FAIL=DINT(FAIL*1000000)/1000000

C      print *,worsteoj,ac,frt(k),fail
C      pause
      FIX=REPAIRT(K)
      Q=AC*QPA(K)
      NOFAIL = Q+STKS
      PO=0

```

```

DO 15 J=1,NOFAIL
X=J
C
C CALLS THE SUBROUTINE THAT CALCULATES THE FAILURE PROBABILITY
C DISTRIBUTION FOR EACH PART
C
CALL BACKORDER(Q,X,FAIL,FIX,BO,STKS)
BORDER(K,J)=BO
PO=PO+BO
15 CONTINUE

C
C THIS CALCULATES THE P(0) VALUE AS A RESULT OF THE FACT THAT THE
C DISTRIBUTION MUST SUM TO ZERO
C

BORDER(K,0)=1/(1+PO)

C THIS STEP MAY BE DELAYED UNTIL FINAL COMP

DO 18 J=1,NOFAIL
BORDER(K,J)=BORDER(K,J)*BORDER(K,0)
18 CONTINUE
C----- PRINT *,BORDER(K,0),BORDER(K,1),BORDER(K,2),BORDER(K,3)
C 1,BORDER(K,4),BORDER(K,5),BORDER(K,6),border(k,7),border(k,8)
C 2,border(k,9)
C----- PAUSE

C END OF FAILURE DIST CALCULATIONS

C
C COMPUTES EQ(3-14)/EQ(3-15) - THIS COMPUTES THE FAILURE
DISTRIBUTION
C OF THE FIRST K UP AND THE LAST M-K DOWN FOR EACH COMPONENT
C

C
C P - THE NUMBER OF AIRCRAFT THAT ARE NOT BROKEN (IE.OPERATING)
C SYSTEMS - THE NUMBER OF COMPONENTS ACROSS ALL SYSTEMS THAT ARE
C OPERATING
C MINFAIL - THE MINIMUM NUMBER OF BACKORDERS THAT MUST OCCUR
C IN ORDER TO CAUSE AC-P AIRCRAFT TO BE BROKEN
C HIGHFAIL - THE MAXIMUM NUMBER OF BACKORDERS THAT CAN OCCUR AND
C STILL ALLOW P AIRCRAFT TO OPERATE
C D - THIS IS THE NUMBER OF BACKORDERS THAT CAN OCCUR BETWEEN
C MINFAIL AND HIGHFAIL
C BOFL - THIS IS THE ACTUAL NUMBER OF FAILURES NECESSARY TO CAUSE
C D BACKORDERS
C ALL OTHER VARIABLES ARE DESCRIBED EARLIER
C

C IF (PASS.EQ.2) THEN
DO 51 P=0,AC

```

```

      U(K,P)=0
51      CONTINUE
C      ENDIF

      DO 65 P=0,AC
        SYSTEMS=AC*QPA(K)
        IF (MINQPA(K).EQ.QPA(K)) THEN
          MINFAIL=(AC-P)
          HIGHFAIL=MINFAIL*QPA(K)
        ELSE
          IF (MINQPA(K).NE.1) THEN
            PRINT *, 'WARNING: THIS PROGRAM ASSUMES THAT IF THERE
IS'
            PRINT *, 'REDUNDANCY THE MIN QPA IS 1. PART',K,' DOES NOT '
            PRINT *, 'HAVE A MIN QPA OF 1.'
            return
          ENDIF
          MINFAIL=(AC-P)*QPA(K)
          HIGHFAIL=P*(QPA(K)-MINQPA(K))+MINFAIL
        ENDIF
        DO 67 D=MINFAIL,HIGHFAIL

          IF (D.EQ.0) THEN
            DO 68 J=0,STKS
              U(K,P)=U(K,P)+BORDER(K,J)
68              CONTINUE
              goto 67
            ENDIF
            BOFL=D+STKS
C            PRINT *,K,P,D,BOFL,MINFAIL,HIGHFAIL
            U(K,P)=U(K,P)+(HYPERT1(MINQPA(K),QPA(K),P,D)*BORDER(K,BOFL))
C            PRINT *,U(K,P),HYPERT1(MINQPA(K),QPA(K),P,D),BORDER(K,BOFL)

67          CONTINUE
65          CONTINUE

C      PAUSE
C
C      PRINTS THE FAILURE DISTRIBUTION FOR EACH COMPONENT FOR THE FIRST
K
C      AIRCRAFT UP AND THE LAST M-K AIRCRAFT DOWN
C
      EOJ=0
      IF (PASS.GT.1.AND.WORST(I).EQ.K) THEN
        I=I+1
      ELSE
C      J - NUMBER OF A/C DOWN-U(K,J) - PROB ARE J UP
        DO 60 J=0,AC
          EOJ=EOJ+(HYPER(AC,AC-J)*U(K,AC-J)*(AC-J))
C----- PRINT *,J,U(K,J),EOJ
60      CONTINUE
          IF (EOJ.LT.RORSTEOJ) THEN
            RORST=K
            RORSTEOJ=EOJ
          ENDIF

```

```

C-----      PRINT *,J,U(K,J),EOJ,RORST,RORSTEOJ
C-----      PAUSE

                ENDIF

                UTOT(AC)=UTOT(AC)*U(K,AC)

25  CONTINUE

        WORSTE=RORSTEOJ
C-----      print *,worste,rorstEOJ

        IF (PASS.EQ.1) THEN
                WORST(1)=RORST
                WORSTEOJ(1)=RORSTEOJ
        ELSE
                DO 604 J=1,PASS-1
                        IF (RORST.LT.WORST(J)) THEN
                                DO 607 K=PASS,J+1,-1
                                        WORST(K)=WORST(K-1)
                                        WORSTEOJ(K)=WORSTEOJ(K-1)
                                607      CONTINUE
                                        WORST(J)=RORST
                                        WORSTEOJ(J)=RORSTEOJ
                                        GOTO 606
                                ENDIF
                        604  CONTINUE
                IF (J.EQ.PASS) THEN
                        WORSTEOJ(J)=RORSTEOJ
                        WORST(J)=RORST
                ENDIF

                ENDIF

                PASS=PASS+1
                CHANGE=UTOT(AC)-CHANGE
C-----      PRINT *, 'pass',PASS,worste,change
C-----      PAUSE

                IF (PASS.GT.30) GOTO 617
                IF (CHANGE.GT.0.001) THEN
                        CHANGE=UTOT(AC)
                        I=1
                        IF (PASS.LT.LRU) THEN
                                GOTO 24
                        ENDIF
                ENDIF

                617  CONTINUE

C-----      PRINT *, 'pass',PASS,worste
C          DO 614 J=1,30
C          PRINT *,J,WORST(J),WORSTEOJ(J)
C 614  CONTINUE
C          PAUSE

```

```

C----- only looks at all/all-1 o.w. upac=0 for all combs
      upac=ac-1
      lrucnt=lru
C      print *, 'u(1,3),u(1,2)',u(1,3),u(1,2)
C      print *, 'u(2,3),u(2,2)',u(2,3),u(2,2)
      do 500 i=1,trun
      if (lrucnt.gt.batch) then
        do 510 j=1,batch
          do 520 k=upac,ac
            v(j,k)=u(j+(i-1)*batch,k)
C            print *, 'i,v(j,k),j,k,u(j+(i-1)*batch,k) '
C            print *, i,v(j,k),j,k,u(j+(i-1)*batch,k)
520          continue
510        continue
        lrupass=batch
      else
        do 530 j=1,lrucnt
          do 540 k=upac,ac
            v(j,k)=u(j+(i-1)*batch,k)
C            print *, 'i,v(j,k),j,k,u(j+(i-1)*batch,k),pt2,u(2,2),u(2,3) '
C            print *, i,v(j,k),j,k,u(j+(i-1)*batch,k), 'pt2',u(2,2),u(2,3)
540          continue
530        continue
        lrupass=lrucnt
      endif
      CALL UPAIRCFT(LRUpass,AC,v,UTOT)
      BORDER(i,ac)=utot(ac)
      BORDER(i,upac)=utot(upac)
      lrucnt=lrucnt-batch
500    continue
C      print *, 'trun,border(1,3),border(1,2),border(2,3),border(2,2) '
C      print *, trun,border(1,3),border(1,2),border(2,3),border(2,2)
      call upaircft(trun,ac,BORDER,utot)

      DO 605 I=1,5
        LIMITEOJ(I)=AC
605    CONTINUE

      IF (PASS.GT.5) THEN
        DO 611 I=1,PASS-1
          EOJ=WORSTEOJ(I)
C        PRINT *,EOJ
        DO 618 J=1,5
          IF (EOJ.LT.LIMITEOJ(J)) THEN
            IF (J.GT.1) THEN
              LIMITNSN(J-1)=LIMITNSN(J)
              LIMITEOJ(J-1)=LIMITEOJ(J)
            ENDIF
            LIMITEOJ(J)=EOJ
C          PRINT
          *,I,EOJ,'WORST',INT(WORST(I)), 'NSN',NSN(INT(WORST(I)))

```

```

        LIMITNSN(J)=NSN(WORST(I))
    ELSE
        GOTO 611
    ENDIF
618 CONTINUE
611 CONTINUE

C      DO 705 I=1,5
C      PRINT *,I,LIMITEOJ(I),LIMITNSN(I)
C 705 CONTINUE
C      PAUSE

    ELSE

        DO 615 K=1,LRU
            EOJ=0
C      J - NUMBER OF A/C DOWN-U(K,J) - PROB ARE J UP
        DO 609 J=0,AC
            EOJ=EOJ+(HYPER(AC,AC-J)*U(K,AC-J)*(AC-J))
C----- PRINT *,J,U(K,J),EOJ
        609 CONTINUE

        DO 608 J=1,5
            IF (EOJ.LT.LIMITEOJ(J)) THEN
                IF (J.GT.1) THEN
                    LIMITNSN(J-1)=LIMITNSN(J)
                    LIMITEOJ(J-1)=LIMITEOJ(J)
                ENDIF
                LIMITEOJ(J)=EOJ
                LIMITNSN(J)=NSN(K)
            ELSE
                EXIT
            ENDIF
        608 CONTINUE

        615 CONTINUE
    ENDIF

    avail1=utot(ac)*100
    avail2=(utot(upac)*100*ac)+avail1
    avail3=avail2-avail1

    PRINT 629, PASS-1
    print *, ' '
    print 630, ac,avail1
    print 640, upac,avail2
    print 650, upac,avail3
    WRITE(17,629) PASS-1
629 FORMAT(1X,'This run had ',i3,' cycles through the parts.')
    write(17,*) ' '
    WRITE(17,630) AC,avail1
630 FORMAT(1X,'The availability of ',I3,' systems is ',2P,E18.11)
    WRITE(17,640) upac,avail2
640 FORMAT(1X,'The availability of ',I3,' or more systems is '
1,2P,E18.11)
    WRITE(17,650) upac,avail3

```

```

650  FORMAT(1X,'The availability of exactly',I3,' systems is '
      1,2P,E18.11)
      WRITE(17,*) ' '
      PRINT *,' '
      WRITE(17,651)
      PRINT *,' This is a list of the top 5 problem parts'
651  FORMAT(1X,'This is a list of the top 5 problem parts')
      WRITE(17,652)
      PRINT *,' Stock Number      E[0]'
652  FORMAT(2X,'Stock Number',8x,'E[0]')

      DO 655 J=5,1,-1
        IF (LIMITNSN(J).GT.'      ') THEN
          WRITE(17,656) LIMITNSN(J),LIMITEOJ(J)
          PRINT 656, LIMITNSN(J),LIMITEOJ(J)
        ENDIF
656  FORMAT(1X,A13,5X,1P,E9.3)
655  CONTINUE

comment      CALL TIMER(DELTA)
C      WRITE(17,*) 'THIS IS THE CPU TIME',DELTA

      END

C-----
-
      SUBROUTINE MYSTUFF(AC,HYPER,HYPERT1)

      INTEGER B(0:10),C(10),AC,MINQPA1,UPAC,MINFAIL,HIGHFAIL
      1,SYSTEMS,DISTFAIL,R,brkac
      REAL*8 BOTTOM,HYPERTIM,HYPERTOT,HYPERT1(10,10,0:10,0:40)
      2,HYPER(100,0:100)

C
C      HYPERT1(A,B,C,D) - THIS IS THE HYPERGEOMETRIC COMBINATIONS THAT
C                        ALLOW C UP AIRCRAFT WITH D BACKORDERS BASED ON
C                        A COMPONENT QPA OF B WITH A MINIMUM QPA OF A
C                        OR HYPERT1(MINQPA,QPA,UPAC,BO)
C
C      NEED TO CHANGE HYPERT1 TO COMMON BLOCK IF THIS BECOMES A CALL
C
C      BUILDS HYPER DISTRIBUTION MATRIX

      K=1

C      THIS IS OVER THE EXPECTED NUMBER OF AIRCRAFT FAILED

C      L = NUMBER OF DOWN AIRCRAFT

      DO 200 L=0,AC
        MINQPA1=1
C      M = QPA OF COMPONENT
        DO 210 M=1,3
          UPAC=AC-L
          SYSTEMS=AC*M

```

```

        MINFAIL=L*M
        HIGHFAIL=UPAC*(M-MINQPA1)+MINFAIL
C      N = NUMBER OF BACKORDERS POSSIBLE UNDER L & M CONDITIONS
        DO 220 N=MINFAIL,HIGHFAIL
            DO 223 J=1,AC
                C(J)=0
223          CONTINUE

            DISTFAIL=N-MINFAIL
            IF (DISTFAIL.EQ.0) GOTO 43
225          CONTINUE
            DO 230 R=1,UPAC
                C(R)=C(R)+1
                DISTFAIL=DISTFAIL-1
                IF (DISTFAIL.EQ.0) goto 231
230          CONTINUE
231          IF (DISTFAIL.GT.0) GOTO 225

C      CALCULATE HYPERGEOMETRIC CALCUALTIONS - EQ(3-13)
C      ASSUMES Rj = R

C      C(X) - MATRIX OF INDIVIDUAL A/C FAILURES
C      B(X) - MATRIX OF CATEGORY COUNTS FOR PERMUTATIONS
C      DO - BOTTOM OF HYPER CALCULATION

C      NO OF FAILURES DISTRIBUTED AMONG UP SYSTEMS
C      WAY ITEMS DISTRIBUTED

C      RESET TOTAL
43      HYPERTOT = 1
C      BUILDS TOP OF 3-13
        DO 45 J=1,AC
            HYPERTOT= HYPERTOT * HYPER(M,C(J))
45      CONTINUE
C      BUILDS BOTTOM OF 3-13
        DO=HYPER(SYSTEMS,N)
C      PRINT *, DO,HYPERTOT,' DO HYPERTOT'
        HYPERTOT = HYPERTOT/DO

C      IF THERE ARE NO UP AIRCRAFT, THEN NO OTHER FAILURE COMBOS ARE
        POSSIBLE

        IF (UPAC.EQ.0) THEN
            HYPERT1(MINQPA1,M,UPAC,N) =
HYPER1(MINQPA1,M,UPAC,N)+HYPERTOT
            goto 220
        ENDIF

C      PRINT *,SYSTEMS,N,M,C(1),C(2),C(3),HYPERTOT,'HYPER'
C      PAUSE

C      RESETS CATEGORY COUNTS B(-)
        DO 48 J=0,M-1

```



```

      B(J)=0
48  CONTINUE
C   CALCULATES CATEGORY COUNTS
      DO 55 J=0,UPAC-1
          B(C(J+1))=B(C(J+1))+1
55  CONTINUE
      BOTTOM=1
C   CALCULATES BOTTOM OF PERMUTATIONS
      DO 58 J=0,M-1
          BOTTOM=FACTORIAL(B(J))*BOTTOM
C   PRINT *,BOTTOM, ' BOTTOM'
58  CONTINUE
C   PERMUTATIONS
      HYPERTIM=FACTORIAL(UPAC)/BOTTOM
C   PRINT *,HYPERTIM, 'NO OF TIMES'
C   HYPERGEOMETRIC TOTAL FOR THE SET (UPAC,N,AND ONE COMBINATION OF
T)
      HYPERTOT=HYPERTOT*HYPERTIM
C   PRINT *,HYPERTOT, 'HYPERTOT'
C   PAUSE

C   TOTAL OF SET T - HYPERT1(MINQPA,QPA,UPAC,BO)
      HYPERT1(MINQPA1,M,UPAC,N) = HYPERT1(MINQPA1,M,UPAC,N)+HYPERTOT

      K=K+1
C   PRINT *, 'K',K, 'C-',C(1),C(2),C(3),C(4)
C   PRINT *,MINQPA1,M,UPAC,N
C   PRINT *, 'HY ',HYPERT1(MINQPA1,M,UPAC,N)
C   PAUSE

      DO 260 D=UPAC,2,-1
          IF (C(D).EQ.0) goto 260
          IF (C(D).EQ.M-1)goto 220
          C(D)=C(D)-1
          DO 270 E=1,UPAC
              IF (C(E).LT.2) THEN
                  IF (E.EQ.D) GOTO 220
                  C(E)=C(E)+1
                  GOTO 43
              END IF
270      CONTINUE
260      CONTINUE
220      CONTINUE
C   PRINT *,P
210      CONTINUE
C   PRINT *,Q
200      CONTINUE

C   HYPERT1(MINQPA,QPA,UPAC,BO)

C   DO 280 D=3,7
C   PRINT *, 'D= ',D,HYPERT1(1,3,2,D)
c 280 CONTINUE
C   pause

C   THIS IS OVER THE EXPECTED NUMBER OF AIRCRAFT FAILED

```

```

C      L is the number of broken a/c.

      DO 202 L=0,ac

C      M = QPA OF COMPONENT

      DO 212 M=1,3

          MINQPA1=M
          UPAC=AC-L
          SYSTEMS=AC*M
          MINFAIL=L
          highfail=L*M

C      N = NUMBER OF FAILURES POSSIBLE UNDER L & M CONDITIONS

      DO 222 N=MINFAIL,HIGHFAIL
      DO 224 J=1,L
          C(J)=0
224      CONTINUE

          DISTFAIL=N

226      CONTINUE
C      PRINT *,DISTFAIL
      DO 232 R=1,L
          C(R)=C(R)+1
          DISTFAIL=DISTFAIL-1
          IF (DISTFAIL.EQ.0) goto 233
232      CONTINUE
233      IF (DISTFAIL.GT.0) GOTO 226

C      CALCULATE HYPERGEOMETRIC CALCUALTIONS - EQ(3-13)
C      ASSUMES Rj = R

C      C(X) - MATRIX OF INDIVIDUAL A/C FAILURES
C      B(X) - MATRIX OF CATEGORY COUNTS FOR PERMUTATIONS
C      DO - BOTTOM OF HYPER CALCULATION

C      NO OF FAILURES DISTRIBUTED AMONG UP SYSTEMS
C      WAY ITEMS DISTRIBUTED

      HYPERT1(MINQPA1,M,UPAC,N)=0

C      RESET TOTAL
44      HYPERTOT = 1
C      BUILDS TOP OF 3-13
      DO 46 J=1,L
          HYPERTOT= HYPERTOT * HYPER(M,C(J))
46      CONTINUE
C      BUILDS BOTTOM OF 3-13
      DO=HYPER(SYSTEMS,N)
C      PRINT *, DO,HYPERTOT,' DO HYPERTOT'
      HYPERTOT = HYPERTOT/DO

```

```

C      PRINT *,SYSTEMS,N,M,C(1),C(2),C(3),HYPERTOT,'HYPER'
C      PAUSE

C      RESETS CATEGORY COUNTS B(-)
      DO 49 J=1,M
        B(J)=0
49    CONTINUE
C      CALCULATES CATEGORY COUNTS
      DO 56 J=1,L
        B(C(J))=B(C(J))+1
56    CONTINUE
      BOTTOM=1
C      CALCULATES BOTTOM OF PERMUTATIONS
      DO 59 J=1,M
        BOTTOM=FACTORIAL(B(J))*BOTTOM
C      PRINT *,BOTTOM,' BOTTOM'
59    CONTINUE
C      PERMUTATIONS
      HYPERTIM=FACTORIAL(L)/BOTTOM
C      PRINT *,HYPERTIM,'NO OF TIMES'
C      HYPERGEOMETRIC TOTAL FOR THE SET (UPAC,N,AND ONE COMBINATION OF
T)
      HYPERTOT=HYPERTOT*HYPERTIM
C      PRINT *,HYPERTOT,'HYPERTOT'
C      PAUSE

C      TOTAL OF SET T - HYPERT1(MINQPA,QPA,UPAC,BO)
      HYPERT1(MINQPA1,M,UPAC,N) = HYPERT1(MINQPA1,M,UPAC,N)+HYPERTOT

      K=K+1
C      PRINT *, 'K',K, 'C-',C(1),C(2),C(3),c(4)
C      print *,MINQPA1,M,UPAC,N
C      PRINT *, 'HY ',HYPERT1(MINQPA1,M,UPAC,N)
C      PAUSE

      brkac=1

      DO 261 D=brkac,2,-1
        IF (C(D).EQ.1) goto 261
        IF (C(D).EQ.M)goto 222
        C(D)=C(D)-1
        DO 271 E=1,brkac
          IF (C(E).LT.M) THEN
            IF (E.EQ.D) GOTO 222
            C(E)=C(E)+1
            GOTO 44
          END IF
271      CONTINUE
261      CONTINUE
222      CONTINUE
C      PRINT *,P
212     CONTINUE
C      PRINT *,Q
202     CONTINUE

C      HYPERT1(MINQPA,QPA,UPAC,BO)

```

```

C      DO 282 D=1,3
C      PRINT *, 'D= ', D, HYPERT1(1,1,D,3-D)
C 282  CONTINUE
C      PRINT *, 'D=1 ', HYPERT1(3,3,1,6)
C      PRINT *, 'D=2 ', HYPERT1(3,3,2,2)
C      PRINT *, 'D=3 ', HYPERT1(2,2,2,1)
C      PRINT *, 'D=2 ', HYPERT1(2,2,1,3)
C      PRINT *, 'SEC3'

```

```

C      END OF HYPERGEOMETRIC CALCULATIONS

```

```

      RETURN
      END

```

```

C-----
-

```

```

      SUBROUTINE HYPERGEO(R,HYPER)

```

```

C
C      FILLS THE HYPERGEOMETRIC CALCULATION MATRIX FOR VARIOUS OPTIONS
C      UP TO R OVER R
C

```

```

      REAL*8 HYPER(100,0:100),CO
      INTEGER R

```

```

C      DO 30 I=1,R
C      PRINT *,i
C      DO 40 J=0,R
C      IF (J .GT. I) goto 30

```

```

C
C      CALLS THE COMBINATORIC CALCULATION SUBROUTINE
C

```

```

      CALL COMBIN(I,J,CO)
      HYPER(I,J)=CO
40  CONTINUE
30  CONTINUE
      RETURN
      END

```

```

C-----
-

```

```

      SUBROUTINE COMBIN(TOPQPA,X,CO)
      INTEGER TOPQPA,X,SUB
      REAL*8 CO,SUB1

```

```

C
C      THIS SUBROUTINE CALCULATES THE NUMBER OF COMBINATIONS POSSIBLE
C      WHERE TOPQPA IS THE NUMBER OF DISTINCT OBJECTS AND X IS THE
C      NUMBER'

```

```

C      OF OBJECTS SELECTED AT AT ONE TIME
C

```

```

C      SUB - DIFFERENCE BETWEEN TOP & BOTTOM OF HYPERGEOM

```

```

C   TOPQPA - TOP OF HYPER
C   X - BOTTOM OF HYPER
C   CO - RESULT OF COMB

```

```

SUB=TOPQPA-X
SUB1=FACTORIAL(SUB)
CO=FACTORIAL(X)*SUB1
CO=FACTORIAL(TOPQPA)/CO

```

```

RETURN
END

```

```

C-----
-

```

```

REAL FUNCTION FACTORIAL(N)
INTEGER N
REAL FACTNO

```

```

C
C   THIS IS A FUNCTION THAT COMPUTES THE FACTORIAL BASED ON THE
C   INCOMING VALUE OF N - (IE. N!)
C

```

```

FACTNO=1.000
IF (N.LE.0) GO TO 105
DO 100 I=1,N
FACTNO=FACTNO*I
100 CONTINUE
105 FACTORIAL=FACTNO
RETURN
END

```

```

C-----
-

```

```

SUBROUTINE BACKORDER(Q,X,FAIL,FIX,BO,STKS)

```

```

C
C   THIS COMPUTES THE FAILURE DISTRIBUTION OF A COMPONENT BASED ON
C   EQ(3-12) WHICH USES A FINITE SOURCE QUEUEING ASSUMPTIONS
C
C   STKS - STOCK LEVEL
C   FAIL - PART FAILURE RATE
C   FIX - REPAIR RATE
C   Q - NUMBER OF COMPONENTS INITIALLY OPERATING
C   X - NUMBER OF COMPONENT FAILURES (IE. DISTRIBUTION #)
C   BO - RESULT OF BACKORDER CALCULATION
C   IADJ - NUMBER OF COMPONENTS ACTUALLY OPERATING
C   RATE - UTILIZATION RATE - FAILURE RATE/FIX RATE
C   ADJ - ADJUSTMENT FOR FINITE SOURCE
C

```

```

REAL*8 FAIL,FIX,RATE,BO
INTEGER Q,X,IADJ,STKS

```

```

RATE=FAIL/FIX
C   PRINT *,RATE,X,STKS,Q
BO=(RATE**X)*(Q**X)/FACTORIAL(X)
IF (X.LE.STKS) GO TO 20
BO=(RATE**X)*(Q**STKS)/FACTORIAL(X)

```

```

      IADJ=Q-X+STKS
      ADJ=FACTORIAL(Q)/FACTORIAL(IADJ)
      BO=BO*ADJ
C      PRINT *,BO
20     RETURN
      END

C-----
-
      SUBROUTINE UPAIRCFT(LRU,AC,V,UTOT)

      INTEGER C(100),LRU,L,M,I,J,K,N,AC,upac
      REAL*8 V(100,0:10),UTOT(0:10),STEP,STEP2,STEP3

C
C      THIS CALCULATES EQ(3-16) - THIS IS THE PROBABILITY DISTRIBUTION
C      FOR THE COMBINED SYSTEM(S) WHERE ALL SYSTEM(S) MUST BE OPERATING
C

C---      MAY WANT TO CONSIDER INIZIALIZING ALL THE UTOTs
C      print *, 'lru,ac,V(1,1),V(1,0),V(2,1),V(2,0)'
C      print *, 'lru,ac,V(1,1),V(1,0),V(2,1),V(2,0)'
C      PAUSE
      UPAC=AC-1
      UTOT(UPAC)=0
      UTOT(AC)=1
      DO 80 K=1,LRU
         UTOT(AC)=UTOT(AC)*V(K,AC)
80     CONTINUE
C      print *, 'THIS IS PROB OF ALL/ALL-1 UP',UTOT(AC),UTOT(UPAC)
C      WRITE(17,*) 'THIS IS PROB OF ALL/ALL-1 UP',UTOT(AC),UTOT(UPAC)
C      PAUSE
C      goto 400

C
C      THIS CALCULATES EQ(3-17) - THIS IS THE PROBABILITY DISTRIBUTION
C      FOR THE COMBINED SYSTEM(S) WHERE ALL BUT THE LAST SYSTEM MUST BE
C      OPERATING
C

C----   NEEDS DOCUMENTING
C----   STEP=UTOT(AC)
C      STEP=UTOT(AC)

C      PRINT *, 'GOT TO HERE'
      K=0

      DO 300 I=1,LRU
         DO 310 J=1,I
            C(J)=J
            RESET=1
310        CONTINUE
C      print *, K, 'GOT TO HERE1',I
C      WRITE(17,*) K, 'GOT TO HERE1',I
C      PAUSE

```

```

315  CONTINUE
      DO 320 J=C(I),LRU
C      IF (I.GE.15) PRINT
*,J,C(I),STEP,UTOT(UPAC),V(J,AC),V(J,UPAC)
      IF (J.EQ.C(I)) THEN
C      PRINT *,J,C(I),STEP,UTOT(UPAC),V(J,AC),V(J,UPAC)
      UTOT(UPAC)=STEP*V(J,UPAC)/V(J,AC)+UTOT(UPAC)
      STEP2=STEP*(V(J,UPAC)/V(J,AC))
      STEP=STEP*(V(J,UPAC)/V(J,AC))
      IF (RESET.EQ.1) THEN
        RESET=0
        STEP3=STEP
      ENDIF
      ELSE
        UTOT(UPAC)=STEP*(V(J,UPAC)/V(J,AC))*V(J-1,AC)/V(J-1,UPAC)
1      +UTOT(UPAC)
        STEP=STEP*(V(J,UPAC)/V(J,AC))*V(J-1,AC)/V(J-1,UPAC)
      ENDIF
      K=K+1
C      IF (I.GE.15) PRINT *,K,STEP,STEP2,UTOT(UPAC)
C      PAUSE
320  CONTINUE
C      PRINT *,K,' GOT TO HERE2',I
C      PAUSE
      IF (I.EQ.1) THEN
        STEP=STEP2
        goto 300
      ENDIF
      DO 330 M=I-1,1,-1
        IF (C(1).EQ.LRU-(I-1)) THEN
          STEP=STEP3
          GOTO 300
        ENDIF
        IF (C(M).EQ.LRU-(I-M)) THEN
C          STEP2=STEP3
          goto 330
        ELSE
          N=M
          goto 331
        ENDIF
330  CONTINUE
C      PRINT *,K,'GOT TO HERE3',N,C(N),C(1),C(2),C(3)
C      PAUSE

C      STEP=STEP2*V(C(N),AC)/V(C(N),UPAC)
331  C(N)=C(N)+1
      DO 340 L=N+1,I
        C(L)=C(L-1)+1
340  CONTINUE

      STEP=UTOT(AC)
      DO 335 M=1,I-1
        STEP=STEP*V(C(M),UPAC)/V(C(M),AC)
335  CONTINUE

      GO TO 315

```

```

C      PRINT *,K,'GOT TO HERE4',C(1),C(2),C(3)
C      PAUSE
300  CONTINUE
C      WRITE(17,*) K,STEP,STEP2,UTOT(UPAC),UTOT(AC)
c 400  continue
C      WRITE(17,*) 'THIS IS THE NUMBER OF COMPs',K
C----- WRITE(17,660) UTOT(AC)
C----- 660  FORMAT(1X,'THIS IS THE PROBABILITY OF ALL SYSTEMS UP - '
C-----      1,2P,E18.11)
C----- WRITE(17,670) UPAC,UTOT(UPAC)
C----- 670  FORMAT(1X,'THIS IS THE PROBABILITY OF ',i3,' SYSTEMS UP -
C-----      1,2P,E18.11)
C----- WRITE(17,*) ' '
C      print *, 'THIS IS THE NUMBER OF COMPs',K
C----- print 660, UTOT(AC)
C----- print 670, upac,UTOT(UPAC)
C----- PRINT *,' '
      RETURN
      END

C -----
-----
      SUBROUTINE CPUTIME(CPTIME)
      REAL*4 CPTIME
comment      TYPE TB_TYPE
comment      SEQUENCE
      REAL*4 USRTIME
      REAL*4 SYSTIME
comment      END TYPE
comment      TYPE (TB_TYPE) DTIME_SRC
      CPTIME=DTIME_(DTIME_SRC)
C      write(6,*)' CPTIME ', CPTIME
      RETURN
      END

C -----
-----
      SUBROUTINE TIMER(DELTA)
      REAL*4 DELTA, CPU2
      CALL CPUTIME( CPU2)
      DELTA = CPU2
      RETURN
      END

C -----
-----

```


The following are the changes and additions that must be made to allow the above code to function as a part of the optimization tool discussed in section 3.5 and 4.4.

```

SUBROUTINE COMP(PASS2,STEP,AVAIL1,AVAIL2,AVAIL3,LIMITNSN
1,LIMITEOJ,FLYHRS, SORTIES,AC)

INTEGER X,Q,STKS,LRUAV,NOFAIL,R,AC,P,BOFL,SRU,QPAS,MINQ,PASS2
1,PASS,WORST(30),LRUPASS,UPAC,LRUCNT,TRUN,BATCH,SRUAV,RORST

FLYHRS=FLYHRS+ (STEP/SORTIES)*PASS2
FLYHRS=DINT(FLYHRS*1E5+0.1)/1E5

IF (PASS2.LT.1) THEN
9 R = 30
CALL HYPERGEO(R,HYPER)
C DO 17 A=1,R
C PRINT *,
HYPER(A,0),HYPER(A,1),HYPER(A,2),HYPER(A,3),HYPER(A,4)
C 17 CONTINUE
C PAUSE

C
C THIS CALLS THE SUBROUTINE THAT CALCULATES THE HYPERGEOMETRIC
C DISTRIBUTION FOR EACH COMBINATION OF MINIMUM QPA, QPA, NUMBER
C OF UP AIRCRAFT AND THE NUMBER OF COMPONENT BACKORDERS
C

CALL MYSTUFF(AC,HYPER,HYPERT1)

WRITE(17,600) AC
600 FORMAT(1X,'THIS IS THE NUMBER OF AIRCRAFT',I3)
WRITE(17,610) LRU
610 FORMAT(1X,'THIS IS THE NUMBER OF LRUS ANALYZED',I4)
WRITE(17,620) SRU
620 FORMAT(1X,'THIS IS THE NUMBER OF SRUS ANALYZED',I4)
WRITE(17,*) ' '

PRINT 600, AC
PRINT 610, LRU
PRINT 620, SRU
PRINT *, ' '
ENDIF

```

APPENDIX B
AIRCRAFT DATA

The input data for B-52H

Stock Number	Work Unit Code	Nomenclature	Demand Rate	Repair Time	Stock Levels		
					(# of A/C)		
					5	3	1
1280001866298	77DCA	ELECTRONIC	0.00202	2.0	0	0	0
1280011207217	73LH0	CONTROL, CO	0.0004	2.0	0	0	0
1280011513174	73QB0	CONVERTER,	0.00023	2.0	0	0	0
1280012088417	75AAJ	INTERVALOM	0.0003	2.0	0	0	0
1280012270719	77EG0	CONTROL UN	0.00081	2.0	1	1	1
1280012283930	73AH0	RECEIVER-T	0.00459	6.0	1	1	1
1280012285349	73AAA	ELECTRONIC	0.00008	5.0	0	0	0
1560005926437	11DBD	DOOR, AIRCR	0.0002	2.0	0	0	0
1560006059700	14EBA	GEAR UNIT	0.00009	2.0	0	0	0
1560008976850	14AKA	SPOILER, WI	0.00058	2.0	0	0	0
1560008976853	14AKA	SPOILER, WI	0.00046	2.0	0	0	0
1560008976866	14AKA	FLAP, WING	0.0002	2.0	0	0	0
1560008978397	14AKA	FLAP, WING	0.00014	2.0	0	0	0
1560010598767	11EMA	DOOR, AIRCR	0.00006	2.0	0	0	0
1660004098966	41KAF	CONTROLBOX	0.00023	2.0	0	1	1
1660011591246	45FAB	VALVE ASSE	0.0002	2.0	0	1	1
1660011680330	41NAP	CONTROL BO	0.00035	2.0	0	0	0
4320006847073	46EAA	PUMP UNIT,	0.00013	2.0	0	1	0
4810008125687	41GAZ	VALVE, BUTT	0.00127	5.0	1	1	1
5821012287058	63AK0	RECEIVER-T	0.00237	2.0	1	1	1
5841010781344	73MA0	RECEIVER-T	0.0013	2.0	0	0	0
5865010887202	76TB0	RECEIVER-T	0.0107	2.0	1	1	1
5895000395486	76WC0	INDICATOR,	0.0067	2.0	1	1	1
5895009049816	76WE0	AMPLIFIER,	0.00101	2.0	1	1	1
5895010620002	63GG0	MODEM, COMM	0.00005	2.0	0	0	0
5895010701444	63GH0	CONTROL, IN	0.00026	2.0	0	0	0
5895011327476	76NB0	DUPLEXER	0.00077	2.0	0	0	0
5895011525214	76EF0	CONTROL-IN	0.00072	2.0	0	0	0
6220011987217	44AFA	LIGHT, TAXI	0.0004	2.0	1	1	1
6340008117801	49DDA	CONTROL, AL	0.00031	2.0	1	1	1
6605012094229	73PC0	BATTERY AN	0.00489	2.0	3	3	2
6610005300026	51ACA	INDICATOR,	0.00065	2.0	1	1	1
6610010456372	51ANK	INDICATOR,	0.00168	2.0	1	1	1
6615005568510	51ANE	GYROSCOPE,	0.00003	2.0	0	0	0
6615007220964	52EE0	CONTROLLER	0.00055	4.0	1	1	1
6615011655941	14FND	CONTROL UN	0.00254	2.0	0	1	0
6615012253746	52ED0	ELECTRONIC	0.00133	2.0	1	1	1
6625004713174	65BDA	TEST SET, T	0.00087	2.0	1	1	1
6680011462360	51BBK	INDICATOR,	0.00031	2.0	0	0	0
6680011469527	51BBJ	INDICATOR,	0.00035	2.0	0	1	1

The input data for E-3B

Stock Number	Work Unit Code	Nomenclature	Demand Rate	Repair Time	Stock Levels (# of A/C)
					5 3 1
1630010098617	13DHE	VALVE, BRAK	0.00011	5.0	0 1 1
1650000773460	46DNB	CYLINDER A	0.00011	6.0	0 1 1
1650010058711	14AGJ	VALVE ASSE	0.00008	2.0	0 0 0
1660001952729	47AKO	REGULATOR,	0.00178	3.0	1 2 1
1660010065617	41BLO	CONTROLLER	0.0003	6.0	0 1 1
1660012756785	41CCO	CONTROLLER	0.00124	2.0	0 0 1
1680010248690	49ABN	MODULE ASS	0.00007	2.0	0 0 0
1680010334612	46FEO	ACTUATOR A	0.00008	2.0	0 0 0
4140010375364	41DBO	FAN, VANEAX	0.00021	2.0	0 0 0
5821011040140	61ANR	CHASSIS, EL	0.00076	3.0	0 0 1
5821011095573	61GAO	CONTROL UN	0.0004	6.0	1 1 1
5826010124864	71KBO	ADAPTER, RE	0.00011	5.0	0 1 1
5841010320176	81BAY	FILTER ASS	0.00027	6.0	0 0 0
5841010367165	81CLE	MODULE	0.00011	5.0	0 0 1
5841010421504	81EJJ	GENERATOR	0.00004	5.0	0 0 0
5841010797775	81QGP	AMPLIFIER-	0.00065	6.0	0 0 0
5841010918929	72AAO	RECEIVER-T	0.00223	5.0	1 1 1
5841011187007	81BMO	MONITOR, RA	0.00041	6.0	0 0 0
5841011287618	81ACK	ELECTRONIC	0.00041	4.0	0 0 1
5841012735169	41HDF	PANEL ASSE	0.00049	3.0	0 0 0
5895010596612	63CKO	AMPLIFIER,	0.00038	5.0	1 1 1
5895011332474	64BAO	CONTROL-IN	0.00041	4.0	0 0 0
5895012511203	69GLP	AMPLIFIER-	0.00159	6.0	0 0 0
5895012580270	69HDO	CONTROL PR	0.00023	5.0	0 0 0
5950010732281	81BAA	TRANSFORME	0.00038	5.0	0 0 0
5960010949540	81BFH	ELECTRON T	0.00041	2.0	0 0 0
5985012297705	69GLA	COUPLER, AN	0.00057	2.0	0 0 0
5985013238387	63DBE	COUPLER, AN	0.00011	2.0	0 0 0
5998013359784	81EJD	CIRCUIT CA	0.0002	5.0	0 0 0
5999010830474	82DAD	CIRCUIT CA	0.00002	6.0	0 0 0
6110002564309	24AJK	CONTROL, VO	0.00023	6.0	0 1 1
6130010127039	42ACO	INVERTER, P	0.00019	2.0	0 0 0
6130012572837	69GKH	POWER SUPP	0.00038	2.0	0 0 0
6610011636927	51KCO	INDICATOR,	0.0003	5.0	0 1 1
6615000228011	51CCO	ADAPTER, PO	0.00537	2.0	1 1 1
6615000593851	51CA0	GYROSCOPE,	0.00015	4.0	0 0 0
6615012107853	52AGO	AMPLIFIER-	0.00095	6.0	0 1 1
6685010179011	41KAO	CONTROL, WI	0.00021	5.0	0 1 1
6685010437538	51HBO	INDICATOR,	0.00038	2.0	0 0 1
6685010621153	45ANO	TRANSMITTE	0.00065	5.0	1 1 1

The input data for F-15C

Stock Number	Work Unit Code	Nomenclature	Demand Rate	Repair Time	Stock Levels		
					(# of A/C)		
					5	3	1
1005000566753	75HD0	GUN, AUTOMA	0.00056	4.0	0	0	0
1005002790528	75HJG	UNIT ASSEM	0.00193	6.0	1	1	1
1620011671000	13AA0	LANDING GE	0.00019	6.0	0	0	0
1650004330145	24DAB	MANIFOLD, H	0.00074	6.0	1	1	1
1650011216981	11PAJ	SERVOCYLIN	0.0005	4.0	1	1	1
1660002876868	41AAL	VALVE, MODU	0.00051	3.0	1	1	1
1680003141930	49ACA	CONTAINER,	0.00193	6.0	2	2	2
1680011417358	46DAP	RECEPTACLE	0.00019	6.0	0	0	0
2835013307765	24BBO	GEARBOX, AC	0.00125	6.0	1	1	1
2915010653149	46AAL	VALVE, POPP	0.001	5.0	1	1	1
4320012855844	45CF0	RESERVOIR,	0.00081	5.0	1	1	1
4320012863686	45AD0	RESERVOIR,	0.00076	5.0	1	1	1
4810003035851	13BEC	VALVE, LINE	0.00006	5.0	1	1	1
4810010214822	41AEB	VALVE, AIR	0.00116	5.0	1	1	1
5810000613388	65BB0	KIR1ATSEC	0.00212	4.0	1	1	1
5841011007363	74FA0	TRANSMITTE	0.00935	4.0	0	0	0
5865011003768	76HB0	OSCILLATIN	0.00302	6.0	0	1	1
5865012876182	76BB0	RECEIVER, C	0.00504	6.0	0	0	0
5865012905835	76BA0	DATA CONVE	0.00462	5.0	0	1	1
5895010959593	63BH0	CONTROL PA	0.00181	3.0	1	1	1
5895011736012	76KA0	PROGRAMMER	0.00247	4.0	1	1	1
5895012278102	74MA0	CONTROL-IN	0.00192	4.0	0	1	1
5895012731990	65BD0	RECEIVER-T	0.00529	5.0	1	1	1
5985010630855	74FU0	ANTENNA	0.00877	4.0	0	0	1
5985012355120	76AK0	ANTENNA	0.00166	6.0	2	1	1
5985012355121	76AN0	ANTENNA	0.00064	4.0	1	1	1
5999012372332	65BAF	CIRCUIT CA	0.00126	5.0	2	1	1
6130010353900	51EAS	POWER SUPP	0.00019	2.0	1	1	1
6605013445803	51NA0	INDICATOR,	0.00901	3.0	1	1	1
6610001342259	51AF0	INDICATOR,	0.00083	4.0	1	1	1
6610001600905	51AJ0	INDICATOR,	0.0009	5.0	1	1	1
6610002963574	51AG0	INDICATOR,	0.0005	5.0	1	1	1
6610003293495	51AH0	ALTIMETER,	0.00081	4.0	1	1	1
6610005357722	51ED0	TRANSMITTE	0.00089	4.0	1	1	1
6610011676617	51AD0	INDICATOR,	0.00228	4.0	1	1	1
6615001377514	52AC0	SENSOR ASS	0.00212	5.0	1	1	1
6615003036730	71FC0	CONTROLLER	0.00055	5.0	1	1	1
6620010344539	46EDB	TRANSMITTE	0.00075	3.0	1	1	1
6680011066215	46EBB	SIGNAL CON	0.0009	6.0	1	1	1
6685010482889	231AB	INDICATOR,	0.00126	4.0	1	1	1

The input data for F-16C

Stock Number	Work Unit Code	Nomenclature	Demand Rate	Repair Time	Stock Levels (# of A/C)		
					5	3	1
1005010418667	75ADA	DRIVE ASSE	0.0005	3.0	1	1	1
1005010446174	75ABC	AMMUNITION	0.00226	1.0	0	0	0
1005010463536	75ACA	UNIT, TRANS	0.00226	3.0	1	1	1
1005010502735	75ABA	AMMUNITION	0.00226	2.0	2	2	2
1005010502736	75ABB	AMMUNITION	0.00226	2.0	2	1	1
1005010556484	75ACE	ACCESS UNI	0.00226	2.0	2	2	2
1270012330011	74AN0	RECEIVER-T	0.00227	6.0	3	2	2
1270012383662	74AP0	TRANSMITTE	0.00268	6.0	4	3	2
1270012566538	74AQ0	PROCESSOR,	0.00328	1.0	3	2	1
1270013093077	74CE0	COMPUTER, F	0.00183	1.0	3	3	2
1270013308895	74BM0	ELECTRONIC	0.00175	1.0	4	3	2
1270013333608	74BT0	SIGHT, HEAD	0.00285	1.0	5	4	2
1290013223711	75DJ0	INTERFACE	0.00189	6.0	3	3	2
1630012523593	13DAA	WHEEL, LAND	0.00637	4.0	6	5	3
1630013201448	13DBA	WHEEL, LAND	0.00325	5.0	2	2	2
1630013826774	13EAH	BRAKE, MULT	0.00268	3.0	2	2	1
1660005678852	47AAA	CONVERTER,	0.00166	2.0	6	5	3
1660011965999	41ADB	CONTROLLER	0.00033	4.0	1	1	1
1660013452115	41AAA	REGULATOR,	0.00047	5.0	1	1	1
2835011156111	24EBA	SHAFT, TURB	0.00127	5.0	1	1	1
2835013083769	24DA0	ENGINE, GAS	0.0005	1.0	0	0	0
2925011150306	24DC0	CONTROLLER	0.0005	1.0	0	0	0
4320013088876	27GJH	PUMP, ROTAR	0.00042	6.0	1	1	1
4320013783398	27GMC	PUMP, ROTAR	0.00045	2.0	0	0	0
4810010996392	24BD0	VALVE, SOLE	0.0005	2.0	0	1	1
4810011307379	24BE0	VALVE, REGU	0.0005	5.0	1	1	1
5810012737820	65AD0	TRANSPONDE	0.0005	6.0	1	1	1
5821012287057	63BL0	RECEIVER-T	0.0031	6.0	7	5	3
5865011106043	76ED0	RECEIVER, C	0.00064	6.0	0	0	0
5865013249103	76EG0	PROCESSOR,	0.00446	6.0	8	6	3
5895010423265	76EA0	CONTROL-IN	0.00021	2.0	1	1	1
5895011074586	76EC0	INDICATOR,	0.00035	6.0	1	1	1
5895011435443	74JA0	DATADISPLA	0.00045	6.0	0	1	0
6340011538696	64AD0	CONTROL, AL	0.00026	5.0	1	1	1
6340013102536	271FC	DETECTOR, I	0.00024	4.0	1	1	1
6605012562380	74DF0	NAVIGATION	0.00417	4.0	6	5	3
6615013619746	14AP0	COMPUTER, F	0.00361	1.0	6	4	3
6620012199413	241AA	INDICATOR,	0.00019	2.0	0	0	0
6620012788027	46ED0	INDICATOR,	0.00045	6.0	1	1	1
6695012305978	14ABC	TRANSDUCER	0.00025	1.0	0	0	0

The following is an example of the format layout of a Dyna-METRIC V4.6 file. This is the format for the LRU fields.

its number on the parent SRU. Quantities must be greater than zero. This field is disregarded for parent records (as designated in the indenture mode specifier).

23-27 XXxxx Depot replacement fraction.
Fraction of the part arriving at a depot on its next higher assembly that will be removed and replaced. This field is disregarded for parent records.

29-33 XXxxx CIRF/Tdepot replacement fraction (subSRUs only).
Fraction of the subSRU arriving at a CIRF or a theater depot on its higher SRU assembly that will be removed and replaced.

35-39 XXxxx Unit replacement fraction (subSRUs only).
Fraction of the subSRU on its higher SRU assembly in unit repair that will be removed and replaced.

"LRU Descriptions"

Header record: LRU

Restriction: Must follow the scenario record groups.

General description:

These records describe the failure, repair and resupply characteristics of each LRU. A pair of these records is required for each LRU.

Columns (first record of pair)

1 2 3 4
1234567890123456789012345678901234567890

aaaaaaaaaaaaaaaaaaaa aaaa i iiiiXXXXXX

Onshore demand rate
NRTS/condemnation/failed SRU policy
Demands per mission indicator
Minimum quantity
Quantity per weapon system
CIRF/Tdepot reparability switch
Level of repair
Depot name

LRU name

4 5 6 7
1234567890123456789012345678901234567

XXXXXXXXXXXX XXxx XXXXXXXX XXxx XXxx

CIRF/Tdepot-served unit
condemnation rate
CIRF/Tdepot-served unit NRTS rate
CIRF/Tdepot-served unit repair time
Lone unit condemnation rate
Lone unit NRTS rate
Lone unit repair time
Offshore demand rate

Detailed description:

For LRUs not assigned to constrained repair, repair (cycle) time includes time awaiting maintenance and in work (and time awaiting parts if there are no SRUs). Otherwise, it is the time the repair resource is exclusively dedicated to the LRU (excluding time spent in the queue).

Condemnation rates are given as a percentage of all LRUs, not as a percentage on non-NRTS LRUs. Thus the sum of the condemnation and NRTS rates for a particular echelon cannot exceed one (100%).

Columns (second record of pair)

1 2 3 4
123456789012345678901234567890123456

aaaaaaaaaaaaaaaaa XXxx XXxx XXXxXXXXxx XXXx XXxx

LRU name
Depot condemnation rate
Depot repair limit
Depot repair time
CIRF/Tdepot condemnation rate
CIRF/Tdepot NRTS rate
CIRF/Tdepot repair time

4 5 6 7
78901234567890123456789012345

XXXXxXXXXx XXXXXXXX aaaaaa i

Cost
Work unit code
No cross-substitution indicator
Wartime resupply time
Peacetime resupply time

First record of pair:

Columns Format

1-16 a16

LRU name.

The name of the LRU. This may be any string of 16 characters as long as it neither the name of another LRU, SRU or subSRU nor its first four characters a keyword (such as "BASE1 ENGINE").

18-21 aaaa

Depot name.

The name of the depot that repairs this LRU. Must be named in the DEPT record group. If the LRU is not repaired by a depot, this field must be blank.

23 i

Level of repair.

Set to 1 if the LRU can be repaired at a unit, CIRF, Tdepot or depot. Set to 2 if the LRU can be repaired at a CIRF, Tdepot or depot but not a unit. Set to 3 if the LRU can only be repaired at a depot. Regardless of setting, LRU always experiences unit administrative delay (specified in TOP record group).

25 i CIRF/Tdepot reparability switch.
Set to 1 if CIRFs and Tdepots can repair the LRU;
set to 0 or blank if they cannot. This field is
disregarded if the level of repair is 3 (depot only).

26-28 iii Quantity per weapon system.
Number of this LRU per weapon system at all units.
Different QPAs across units are specified in
the QPA record group and override this field.

29-31 iii Minimum quantity.
Minimum number of this LRU required for the
weapon system to be mission capable (i.e., the QPA
from the previous field less the number that may
be broken without impairing the weapon system's
capability). Different minimum quantities across
units are specified in the QPA record group and
override this field.

32 i Demands per mission indicator.
Set to 1 if the demand rates (described below) are
per mission; set to 0 or blank if the demand rates
are per operating hour. (Mode of failure for
indentured SRUs is the same as that of the LRU.)

33 i NRTS/condemnation/failed SRU policy.
Indicates when the decision to NRTS or condemn the
LRU is made, and when failed SRUs indentured to
the LRU are discovered. Set to 1 if these occur
before attempting repair; set to 0 or blank if the
LRU first enters repair.

34-40 XXXXXXXX Onshore demand rate (peacetime).
At onshore units, the fraction of the LRU that breaks
per operating hour/mission (determined in column 32)
in peacetime. (The wartime demand rate is
determined by multiplying this field by the onshore
demand rate multiplier in the VTM record group.)

41-47 XXXXXXXX Offshore demand rate (peacetime).
At offshore units, the fraction of the LRU that breaks
per operating hour/mission (determined in column 32)
in peacetime. (The wartime demand rate is
determined by multiplying this field by the offshore
demand rate multiplier in the VTM record group.)

48-52 XXXxx Lone unit repair time (in days).
Repair time at units not served by a CIRF or Tdepot.

54-57 XXxx Lone unit NRTS rate.
Fraction of removals at units not served by a
CIRF or Tdepot that are declared NRTS.

59-62 XXxx Lone unit condemnation rate.
Fraction of removals at units not served by a
CIRF or Tdepot that are declared condemned.

63-67 XXXxx CIRF/Tdepot-served unit repair time (in days).
Repair time at units served by a CIRF or Tdepot.

69-72 XXxx CIRF/Tdepot-served unit NRTS rate.
 Fraction of removals at units served by a CIRF
 or Tdepot that are declared NRTS.

74-77 XXxx CIRF/Tdepot-served unit condemnation rate.
 Fraction of removals at units served by a CIRF
 or Tdepot that are declared condemned.

Second record of pair:

Columns Format

1-16 a16 LRU name.
 The name of the LRU. Must match LRU name given
 on first record of pair.

18-21 XXxx CIRF/Tdepot repair time (in days).
 Repair time at CIRFs and Tdepots.

23-26 XXxx CIRF/Tdepot NRTS rate.
 Fraction of removals at CIRFs and Tdepots that are
 declared NRTS.

28-31 XXxx CIRF/Tdepot condemnation rate.
 Fraction of removals at CIRFs and Tdepots that are
 declared condemned.

32-36 XXXxx Depot repair time (in days).
 Repair time at the depot.

38-41 XXXx Depot repair limit.
 Maximum number of the LRU that can be repaired at
 the depot each day during wartime. (Applies only to
 LRUs not assigned to constrained repair.) A value
 of 0 means no limit, while a value of -1 means no
 depot repair.

43-46 XXxx Depot condemnation rate.
 Fraction of removals at the depot that are declared
 condemned.

47-51 XXXXx Peacetime resupply time (in days).
 The expected time for the highest echelon repairing
 the LRU to procure a replacement during peacetime.

52-56 XXXXx Wartime resupply time (in days).
 The expected time for the highest echelon repairing
 the LRU to procure a replacement during wartime.

58-65 XXXXXXXX Cost.
 Unit cost of the LRU.

67-73 aaaaaaa Work unit code.
 This field may contain any string of 7 characters.

75 i No cross-substitution indicator.
 Set to 1 if the LRU cannot be cross-substituted; set
 blank or 0 if it can. This switch affects only the
 results in the performance report (selected with

APPENDIX C
SIMULATION MODEL CODE

This is the SLAM II simulation code used in this research.

This is the network code.

GEN,TLEWIS,DISSERTATION,6/04/95,1,N,N,,Y,Y/1,72;

LIM,50,50,25;

NETWORK;

RESOURCE/LRU1(0),11,3;

RESOURCE/LRU2(0),12,3;

RESOURCE/LRU3(0),13,3;

RESOURCE/LRU4(0),14,3;

RESOURCE/LRU5(0),15,3;

RESOURCE/LRU6(0),16,3;

LRU SPARES AVAILABLE

RESOURCE/LRU7(0),17,3;

RESOURCE/LRU8(0),18,3;

RESOURCE/LRU9(0),19,3;

RESOURCE/LRU10(0),20,3;

RESOURCE/LRU11(0),21,3;

RESOURCE/LRU12(0),22,3;

RESOURCE/LRU13(0),23,3;

RESOURCE/LRU14(0),24,3;

LRU SPARES AVAILABLE

RESOURCE/LRU15(0),25,3;

RESOURCE/LRU16(0),26,3;

RESOURCE/LRU17(0),27,3;

RESOURCE/LRU18(0),28,3;

RESOURCE/LRU19(0),29,3;

RESOURCE/LRU20(0),30,3;

RESOURCE/LRU21(0),31,3;

RESOURCE/LRU22(0),32,3;

RESOURCE/LRU23(0),33,3;

RESOURCE/LRU24(0),34,3;

RESOURCE/LRU25(0),35,3;

LRU SPARES AVAILABLE

RESOURCE/LRU26(0),36,3;

RESOURCE/LRU27(0),37,3;

RESOURCE/LRU28(0),38,3;

RESOURCE/LRU29(0),39,3;

RESOURCE/LRU30(0),40,3;

RESOURCE/LRU31(0),41,3;

RESOURCE/LRU32(0),42,3;

RESOURCE/LRU33(0),43,3;

RESOURCE/LRU34(0),44,3;

RESOURCE/LRU35(0),45,3;

RESOURCE/LRU36(0),46,3;

LRU SPARES AVAILABLE

RESOURCE/LRU37(0),47,3;

RESOURCE/LRU38(0),48,3;

RESOURCE/LRU39(0),49,3;

RESOURCE/LRU40(0),50,3;

RESOURCE/EQ(25),6,5,2;

CREATE,24,0;

ASS,XX(52)=XX(51)*24,XX(51)=XX(51)+1;

EVENT,2; GETS DAILY SCHEDULE

ASS,XX(6)=0,XX(2)=0,XX(5)=NNQ(1)+NNQ(7);

ACT,23.99999;

GOON;

TED EVENT,6;

COLCT,XX(6),SORTIES FLOWN;

```

TE  TERM;

      CREATE,0,0,,24,1;
      ACT,,XX(1).GE.XX(54),TER;
      ACT;      CREATES AIRCRAFT
      ASS,XX(1)=XX(1)+1,XX(5)=XX(54),ATTRIB(5)=XX(1),XX(75)=XX(54);
      ASS,XX(76)=1,XX(77)=1;
      EVENT,4;      ASSEMBLES AIRCRAFT
BEG  ASS,XX(56)=USERF(2),1;
SOR1 ASS,ATTRIB(7)=ATTRIB(7)+1,XX(6)=XX(6)+1;
      ACT/1,EXPON(XX(56),1);      FLYS SORTIE
      ASS,ATTRIB(6)=1;
      GOON,1;
      ACT,,BRK;

BDAY ASS,XX(8)=TNOW-XX(52),1;
      ACT,,BEG;

BRK  COLCT,BETW,TIME BTW FAIL;
      ASSIGN,ATTRIB(1)=TNOW,XX(75)=XX(75)-1,XX(77)=0,1;  MARKS TIME OF
FAILURE
      ACT,,XX(75).GE.XX(54)-1,STB1;
      ACT/8;
      ASS,XX(76)=0;
STB1 AWAIT(2),EQ/1;      WAITS FOR REPAIR MAN/EQUIPMENT
STB  ASS,XX(60)=XX(66)/XX(7),1;
      EVENT,1,1;
      ACT/5,,ATTRIB(8).EQ.0,FIX1; FIX AC
      ACT/2,XX(9);      REMOVES PART

      COLCT,ATTRIB(8),NUMBER OF BREAKS;
      COLCT,ATTRIB(9),SECOND BREAK;

      GOON,1;
      ACT/6,,XX(53).EQ.1,CANN; CANN
      ACT/7,,XX(53).EQ.0; NO CANN
      FREE,EQ/1;
      ACT,,NCAN;
      ACT,,ILVL;

CANN EVENT,3,2;
      ACT/10,,ILVL;
      ACT/11,,ATTRIB(3).EQ.0,FRE;
      ACT/12;

;C NCAN AWAIT(4),ALLOC(2),,1;
NCAN EVENT,8,1;
      ACT,,ATTRIB(2).EQ.0,NCA3;
      ACT,,ATTRIB(2).EQ.1,AW11;
      ACT,,ATTRIB(2).EQ.2,AW12;
      ACT,,ATTRIB(2).EQ.3,AW13;
      ACT,,ATTRIB(2).EQ.4,AW14;
      ACT,,ATTRIB(2).EQ.5,AW15;
      ACT,,ATTRIB(2).EQ.6,AW16;
      ACT,,ATTRIB(2).EQ.7,AW17;
      ACT,,ATTRIB(2).EQ.8,AW18;
      ACT,,ATTRIB(2).EQ.9,AW19;

```

```

ACT,,ATRIB(2).EQ.10,AW20;
ACT,,ATRIB(2).EQ.11,AW21;
ACT,,ATRIB(2).EQ.12,AW22;
ACT,,ATRIB(2).EQ.13,AW23;
ACT,,ATRIB(2).EQ.14,AW24;
ACT,,ATRIB(2).EQ.15,AW25;
ACT,,ATRIB(2).EQ.16,AW26;
ACT,,ATRIB(2).EQ.17,AW27;
ACT,,ATRIB(2).EQ.18,AW28;
ACT,,ATRIB(2).EQ.19,AW29;
ACT,,ATRIB(2).EQ.20,AW30;
ACT,,ATRIB(2).EQ.21,AW31;
ACT,,ATRIB(2).EQ.22,AW32;
ACT,,ATRIB(2).EQ.23,AW33;
ACT,,ATRIB(2).EQ.24,AW34;
ACT,,ATRIB(2).EQ.25,AW35;
ACT,,ATRIB(2).EQ.26,AW36;
ACT,,ATRIB(2).EQ.27,AW37;
ACT,,ATRIB(2).EQ.28,AW38;
ACT,,ATRIB(2).EQ.29,AW39;
ACT,,ATRIB(2).EQ.30,AW40;
ACT,,ATRIB(2).EQ.31,AW41;
ACT,,ATRIB(2).EQ.32,AW42;
ACT,,ATRIB(2).EQ.33,AW43;
ACT,,ATRIB(2).EQ.34,AW44;
ACT,,ATRIB(2).EQ.35,AW45;
ACT,,ATRIB(2).EQ.36,AW46;
ACT,,ATRIB(2).EQ.37,AW47;
ACT,,ATRIB(2).EQ.38,AW48;
ACT,,ATRIB(2).EQ.39,AW49;
ACT,,ATRIB(2).EQ.40,AW50;

```

```

AW11 AWAIT(11),ALLOC(2);
ACT,,NCA2;
AW12 AWAIT(12),ALLOC(2);
ACT,,NCA2;
AW13 AWAIT(13),ALLOC(2);
ACT,,NCA2;
AW14 AWAIT(14),ALLOC(2);
ACT,,NCA2;
AW15 AWAIT(15),ALLOC(2);
ACT,,NCA2;
AW16 AWAIT(16),ALLOC(2);
ACT,,NCA2;
AW17 AWAIT(17),ALLOC(2);
ACT,,NCA2;
AW18 AWAIT(18),ALLOC(2);
ACT,,NCA2;
AW19 AWAIT(19),ALLOC(2);
ACT,,NCA2;
AW20 AWAIT(20),ALLOC(2);
ACT,,NCA2;
AW21 AWAIT(21),ALLOC(2);
ACT,,NCA2;
AW22 AWAIT(22),ALLOC(2);
ACT,,NCA2;
AW23 AWAIT(23),ALLOC(2);

```

```
      ACT,,,NCA2;
AW24  AWAIT(24),ALLOC(2);
      ACT,,,NCA2;
AW25  AWAIT(25),ALLOC(2);
      ACT,,,NCA2;
AW26  AWAIT(26),ALLOC(2);
      ACT,,,NCA2;
AW27  AWAIT(27),ALLOC(2);
      ACT,,,NCA2;
AW28  AWAIT(28),ALLOC(2);
      ACT,,,NCA2;
AW29  AWAIT(29),ALLOC(2);
      ACT,,,NCA2;
AW30  AWAIT(30),ALLOC(2);
      ACT,,,NCA2;
AW31  AWAIT(31),ALLOC(2);
      ACT,,,NCA2;
AW32  AWAIT(32),ALLOC(2);
      ACT,,,NCA2;
AW33  AWAIT(33),ALLOC(2);
      ACT,,,NCA2;
AW34  AWAIT(34),ALLOC(2);
      ACT,,,NCA2;
AW35  AWAIT(35),ALLOC(2);
      ACT,,,NCA2;
AW36  AWAIT(36),ALLOC(2);
      ACT,,,NCA2;
AW37  AWAIT(37),ALLOC(2);
      ACT,,,NCA2;
AW38  AWAIT(38),ALLOC(2);
      ACT,,,NCA2;
AW39  AWAIT(39),ALLOC(2);
      ACT,,,NCA2;
AW40  AWAIT(40),ALLOC(2);
      ACT,,,NCA2;
AW41  AWAIT(41),ALLOC(2);
      ACT,,,NCA2;
AW42  AWAIT(42),ALLOC(2);
      ACT,,,NCA2;
AW43  AWAIT(43),ALLOC(2);
      ACT,,,NCA2;
AW44  AWAIT(44),ALLOC(2);
      ACT,,,NCA2;
AW45  AWAIT(45),ALLOC(2);
      ACT,,,NCA2;
AW46  AWAIT(46),ALLOC(2);
      ACT,,,NCA2;
AW47  AWAIT(47),ALLOC(2);
      ACT,,,NCA2;
AW48  AWAIT(48),ALLOC(2);
      ACT,,,NCA2;
AW49  AWAIT(49),ALLOC(2);
      ACT,,,NCA2;
AW50  AWAIT(50),ALLOC(2);
      ACT,,,NCA2;
```

```

NCA2 COLCT,INT(1),AWP TIME;
    EVENT,9,1;
    ACT,,XX(53).EQ.1,REPR;
    ACT,,XX(53).EQ.0;
NCA3 AWAIT(6),EQ/1;
    ACT,,,REPR;
FRE  FREE,EQ/1;          FREE MAINTENCE MAN/EQUIPMENT
    AWAIT(3),ALLOC(1);
    AWAIT(5),EQ/1;      WAIT FOR MAINTENCE MAN/EQUIPMENT
REPR GOON;
    ACT/3,XX(9);        REPLACE LRU
GO   COLCT,INT(1),REPAIR TIME;
    FREE,EQ/1,1;
    ASS,XX(75)=XX(75)+1,1;
    ACT,,XX(75).LE.XX(54)-2,P1;
    ACT/10,,XX(75).EQ.XX(54),P3;
    ACT/9;
    ASS,XX(76)=1;
    ACT,,,P1;
P3   GOON;
    ASSIGN,XX(77)=1;
P1   GOON,1;
    ACT,,,BDAY;

;C   aircraft repair cycle

FIX1 GOON,1;
    ACT/13,EXPON(XX(59),2),XX(58),TO;
    ACT;
TO   FREE,EQ/1;
    GOON,1;
    ACT,,ATTRIB(1).EQ.TNOW,BDAY;
    ACT;
    ASS,XX(8)=TNOW-ATTRIB(1),1;
    ACT,,XX(8).LT.24,BDAY;
    ACT;
    ASS,ATTRIB(7)=0;
    ACT,,,BDAY;

;C
;C   LRU REPAIR
;C
ILVL GOON;
    ACT,,,EVE;
    ACT,,ATTRIB(9).GT.0;
    ASS,ATTRIB(2)=ATTRIB(9);

;C
;C   CHOOSE BROKEN SRUS AND NRTS RATES
;C
EVE  EVENT,5;
    ASS,ATTRIB(1)=TNOW;
    GOON,1;
    ACT,,XX(57),TER;
    ACT/4,EXPON(ATTRIB(10),2);    REPAIR LRU
    COLCT,INT(1),LRU REPAIR TIME;
    GOON,1;
    ACT,,ATTRIB(9).EQ.0,FR;
    ACT;

```



```

;C
;C      DECIDES IF SRU IS AVAILABLE ?
;C      IF YES, RETURNS LRU TO SUPPLY
;C      IF NOT, THEN CANNIBALIZES SRUS, IF POSSIBLE
;C
      EVENT,7,1;
      ACT,,ATRI(3).EQ.1,FR;
      ACT;
      QUE(8);
      FR  FREE,ATRI(2)/1;  FREE LRU
      TER  TERM;
      END;
INIT,0.,1008000.,N;  RUNS MODEL FOR 30 DAYS
;C MONTR,TRACE,0.0,2500,2,8,11,44,NNRSC(1),NNRSC(34),-55;
TIMST,XX(75),AVAILABILITY,10/0.5/0.5;
TIMST,XX(76),AVAIL2;
TIMST,XX(77),AVAILALL;
SIM;
FIN;

```

This is the user-written FORTRAN subroutines.

```

      program main
      dimension nset(15000)
      common qset(15000)
      common/scom1/
      atrib(100),dd(100),ddl(100),dtnow,ii,mfa,mstop,nclnr

      1,ncrdr,nprnt,nnrun,nnset,ntape,ss(100),ssl(100),tnext,tnow,xx(100)
      equivalence (nset(15000),qset(15000))
      nnset=15000
      ncrdr=5
      nprnt=6
      ntape=7
      call slam
      stop
      end

      SUBROUTINE EVENT(I)

      COMMON/SCOM1/ATRI(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR

      1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/MINE/REPAIRT(40),NRTS(40),QPA(40),RUN,STK(30),SQPA(30)
      1,TLRU(40),SRULOC(40),SRUDM(30),MSOR(130),FMC(130),MQPA(40)
      INTEGER MDAY,DIFF
      C      real avail,avail2
      OPEN(UNIT=18,FILE='OUTPUT.OUT',STATUS='OLD')

      GO TO (1,2,3,4,5,6,7,8,9) I

      C
      C      CHOOSES BROKEN LRU
      C

```

```

1  ATRIB(8)=1
   ATRIB(9)=0
   ATRIB(2)=0
   ATRIB(10)=0
C
C  USES RANDOM DRAWS TO CHOOSE BROKEN LRUS
C  AND THE NUMBER OF DEMANDS
C
   IF (ATLIB(6).LT.1) GO TO 17
   ATRIB(6)=ATLIB(6)-1
   DRAW=UNFRM(0.0,1.0,3)
   IF (DRAW.LT.ATLIB(6)) THEN
   ATRIB(8)=2
   END IF
17  DRAW=UNFRM(0.0,1.0,4)
   IF (DRAW.LT.XX(60)) THEN
   IF (ATLIB(8).EQ.1.0) THEN
   ATRIB(8)=0
   GO TO 21
   END IF
C
C  GETS THE REPAIR TIME
C
   ATRIB(10)=EXPON(XX(59),5)
   ATRIB(8)=ATLIB(8)-1
   GO TO 17
   END IF
15  DRAW=UNFRM(0.0,1.0,5)
   J=INT(10+XX(55))
   DO 10 I=11,J
   IF (XX(I).GE.DRAW) GO TO 20
10  CONTINUE
20  IF(ATLIB(2).EQ.0.0) THEN
C
C  MARKS THE BROKEN LRUS
C
   ATRIB(2)=I-10
   ELSE
   ATRIB(9)=I-10
   END IF

   I=INT(ATLIB(2))
   ATRIB(I+10)=ATLIB(I+10)-1
C
C  DECIDES WHERE THE REPAIR IS TO TAKE PLACE
C
   IF(ATLIB(8).GT.1.0) THEN
   ATRIB(8)=1
   DRAW=UNFRM(0.0,1.0,4)
   IF (DRAW.LT.XX(60)) GO TO 21
   GO TO 15
   END IF
   IF (ATLIB(9).EQ.ATLIB(2)) THEN
   LRU=INT(ATLIB(2))
   IF (QPA(LRU).GT.1) GO TO 21
   GO TO 15

```

```

        END IF
21    RETURN

C
C    FLIGHT SCHEDULE
C
C 2    IF (XX(51).GE.1058) GO TO 22
      2    XX(3)=XX(70)
          XX(4)=XX(71)
          GO TO 24
      22    XX(3)=1.1
          XX(4)=1.7

C
C    GETS DAILY SORTIE REQUIREMENT
C
      24    XX(61)=XX(54)*XX(3)
          ROUND=XX(61)-INT(XX(61))
          IF (ROUND.LT.0.5) XX(61)=INT(XX(61))
          RETURN

C
C    ARE THE BROKEN LRUS AVAILABLE?
C
      3    ATRIB(3)=0
          LRU1=INT(ATRIB(2))
          LRU2=INT(ATRIB(9))
          IF (LRU1.EQ.LRU2) THEN
          IF (NNRSC(LRU1).GT.1) GO TO 30
          GO TO 35
          END IF

C
C    IF YES, GET LRUS
C
          IF (NNRSC(LRU1).GT.0) THEN
          IF (LRU2.LE.0) GO TO 30
          IF (NNRSC(LRU2).GT.0) GO TO 30
          END IF
      35    J=INT(XX(55))

C
C    IF THE BROKEN LRUS ARE NOT AVAILABLE, MAKE THE A/CC
C    LRUS AVAILABLE TO OTHER AIRCRAFT
C
          DO 25 I=1,J
          LRUQPA=INT(QPA(I))
          IF (I.EQ.ATRIB(2)) LRUQPA=LRUQPA-1
          IF (I.EQ.ATRIB(9)) LRUQPA=LRUQPA-1
          CALL FREE(I,LRUQPA)
      25    CONTINUE
          RETURN
      30    ATRIB(3)=1
          RETURN

C
C    ASSEMBLES AIRCRAFT AT BEGINNING OF MODEL
C
      4    J=INT(XX(55))

```

```

DO 40 I=1,J
  LRUQPA=INT(QPA(I))
  ATRIB(I+10)=LRUQPA
  CALL SEIZE(I,LRUQPA)
40 CONTINUE
  RETURN

C
C   DECIDES IF THERE IS A BROKEN SRU?
C   IF YES, IT CHOOSES ONE
C

5   I=INT(ATRIB(2))
    IF (TLRU(I).LE.0) GO TO 53

    DRAW=UNFRM(0.0,1.0,6)
    N=INT(SRULOC(I))
    L=INT(SRULOC(I)+TLRU(I)-1)
    DO 52 M=N,L
      IF (SRUDM(M).GE.DRAW) THEN
        ATRIB(9)=M
        GO TO 55
      END IF
52 CONTINUE
53 ATRIB(9)=0

C
C   GETS NRTS RATE AND REPAIR TIME
C

55  XX(57)=NRTS(I)
    ATRIB(10)=REPAIRT(I)
    RETURN

C
C   CALCULATES OUTPUT STATISTICS
C

6   XX(5)=(XX(5)+NNQ(1)+NNQ(7))/2
    MDAY=INT(XX(51))
C   FMC(MDAY)=FMC(MDAY)+XX(5)
C   MSOR(MDAY)=MSOR(MDAY)+XX(6)
C   WRITE(*,*)RUN,MDAY,XX(6),XX(5),FMC(MDAY),MSOR(MDAY)
C   WRITE(8,*)RUN,MDAY,XX(6),XX(5)
    XX(78)=XX(78)+1
C   write(18,*) xx(78),run,mday,ttavg(1),ttavg(2),xx(75)
    IF (XX(78).GE.400) THEN
      WRITE(18,*) ttavg(1),ttavg(2),ttavg(3),NNCNT(8),NNCNT(9)
C     print *, 'this 1',mday,xx(7),ccavg(3)
      if (mday.ge.42000) call sumry
      CALL CLEAR
C     print *, 'this 2',mday,xx(7),ccavg(3)
      if (mday.ge.1100) call sumry
C     xx(78)=0
C   WRITE(18,*) ttavg(1),ttavg(2),ccavg(3),mday
      XX(78)=0
    ENDIF
    RETURN

C

```

```

C      IS THE BROKEN SRU AVAILABLE?
C      IF YES, FIX LRU
C      IF NO, CANNIBALIZE THE LRU, IF POSSIBLE
C
7      ATRIB(3)=0
      ISRU=INT(ATRIB(9))
      IF (STK(ISRU).GT.0) THEN
      ATRIB(3)=1
      STK(ISRU)=STK(ISRU)-1
      GO TO 75
      END IF
      NLRU=INT(ATRIB(2))
      N=INT(SRULOC(NLRU)+TLRU(NLRU)-1)
      L=INT(SRULOC(NLRU))
      DO 72 I=L,N
      SRUQPA=INT(SQPA(I))
      IF(ATRIB(9).EQ.I) SRUQPA=SRUQPA-1
      STK(I)=STK(I)+SRUQPA
72     CONTINUE
75     RETURN

8      J=INT(XX(55))
      DO 80 I=1,J
      IF (ATRIB(I+10).LT.QPA(I)) THEN
      IF (NNRSC(I).GT.0) THEN
      DIFF=INT(QPA(I)-ATRIB(I+10))
C      write(18,*) diff, qpa(i), mqp(i), nnrsc(i), atrib(2), ATRIB(I+10)
      IF (DIFF.GT.NNRSC(I)) DIFF=INT(NNRSC(I))
      IF (DIFF.LE.0) GOTO 80
      CALL SEIZE(I,DIFF)
      ATRIB(I+10)=ATRIB(I+10)+DIFF
      ENDIF
      ENDIF
80     CONTINUE

      I=INT(ATRIB(2))
C      write(18,*) atrib(2),NNRSC(I),QPA(I),MQPA(I),ATRIB(I+10)
      IF (ATRIB(I+10).LT.MQPA(I)) RETURN
      ATRIB(2)=0
C      write(18,*) atrib(2)
      RETURN

9      J=INT(XX(55))
      I=INT(ATRIB(2))
      ATRIB(I+10)=ATRIB(I+10)+1
      DO 90 I=1,J
      IF (ATRIB(I+10).LT.QPA(I)) THEN
      IF (NNRSC(I).GT.0) THEN
      DIFF=INT(QPA(I)-ATRIB(I+10))
      IF (DIFF.GT.NNRSC(I)) DIFF=INT(NNRSC(I))
      IF (DIFF.LE.0) GOTO 90
      CALL SEIZE(I,DIFF)
      ATRIB(I+10)=ATRIB(I+10)+DIFF
      ENDIF
      ENDIF

```

```

90  CONTINUE
    RETURN

```

```

    END

```

```

    SUBROUTINE INTLC

```

```

COMMON/SCOM1/ATTRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR

```

```

1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
COMMON/MINE/REPAIRT(40),NRTS(40),QPA(40),RUN,STK(30),SQPA(30)
1,TLRU(40),SRULOC(40),SRUDM(30),MSOR(130),FMC(130),MQPA(40)
CHARACTER*13 NUM
INTEGER TRUN,LRUAV,QPAS,MINQ,SRUAV
OPEN(UNIT=17,FILE='FRED.INP',STATUS='OLD')
OPEN(UNIT=18,FILE='SRU.INP',STATUS='OLD')

```

```

C
C  READS IN INITIAL VALUES AND FAILURE RATES
C

```

```

    IF (TRUN.EQ.5) GO TO 8
    TRUN=5
    RUN=0
8   RUN=INT(RUN+1)
    ISRU=0
    AM=0
    IF (RUN.GT.1.1) GO TO 6
    DO 4 I=1,30
    MSOR(I)=0
    FMC(I)=0
4   CONTINUE
6   XX(58)=0.0
    XX(66)=0.00
    XX(9)=0
    READ(17,*) AC
    XX(54)=AC
    READ(17,*) BR
    XX(7)=BR
    READ(17,*) LRU
    XX(55)=LRU
    READ(17,*) SORT
    XX(70)=SORT
    READ(17,*) FLYH
    XX(71)=FLYH
    READ(17,*) CANN
    XX(53)=CANN
    DO 10 I=1,LRU

```

```

C
C  READS IN THE LRU PART NO., DEMAND RATE, QPA, NO. OF LRUS
C  IN STOCK, NO. OF SRUS IN LRU, NRTS RATE, REPAIR CYCLE TIME
C

```

```

    READ(17,7) NUM,FR,QPAS,MINQ,LRUAV,SRUAV,RTS,REP
7   FORMAT(A13,F8.5,1X,I3,1X,I3,1X,I3,I3,F5.2,F5.2)
    TLRU(I)=SRUAV
    IF(SRUAV.EQ.0) GO TO 5
    ISRU1=ISRU+1

```

```

        ISRU=INT(ISRU+SRUAV)
        SRULOC(I)=ISRU1
        TOTAL=0
C
C      READS IN SRU DATA
C      SRU NO., DEMAND RATE, QPA, NO. OF SRUS IN STOCK
C
        DO 2 J=ISRU1,ISRU
        READ(18,*)NU,DM,QPSRU,SRUSTK
        SQPA(J)=QPSRU
        STK(J)=SRUSTK
        TOTAL=TOTAL+DM
        SRUDM(J)=TOTAL/FR
    2    CONTINUE
C
C      BUILDS THE CUMULATIVE FAILURE DISTRIBUTIONS
C
    5    NRTS(I)=RTS
        REPAIRT(I)=REP*24
        MQPA(I)=MINQ
        QPA(I)=QPAS
        LAV=INT(LRUAV+(XX(54)*QPA(I)))
        L=I+10
        XX(L)=AM+(FR*QPA(I))
        AM=XX(L)
        CALL ALTER(I,LAV)
    10   CONTINUE
C      print *,xx(10+int(lru)),xx(7)
        DO 20 I=1,LRU
        L=I+10
        M=10+INT(LRU)
        XX(L)=XX(L)/XX(M)
    20   CONTINUE
        XX(77)=0
        close(17)
        close(18)

        RETURN
        END

        SUBROUTINE OPUT

COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
COMMON/MINE/REPAIRT(40),NRTS(40),QPA(40),RUN,STK(30),SQPA(30)
1,TLRU(40),SRULOC(40),SRUDM(30),MSOR(130),FMC(130),MQPA(40)
OPEN(UNIT=19,FILE='SAS.DAT',STATUS='OLD')
C
C      WRITES OUTPUT STATISTICS
C
        write(19,*)
ttavg(1),ttavg(2),ttavg(3),nnent(8),nnent(9),nnent(10)
        DO 10 I=1,30
        AVGFMC=FMC(I)/RUN
        AVGSOR=MSOR(I)/RUN
        IF (RUN.LT.50) GO TO 20

```

```

C      WRITE(19,*)I,AVGFMC,AVGSOR
10     CONTINUE
C      CLOSE(19)
20     RETURN
      END
      SUBROUTINE ALLOC(I,IFLAG)

COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR

1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
COMMON/MINE/REPAIRT(40),NRTS(40),QPA(40),RUN,STK(30),SQPA(30)
1,TLRU(40),SRULOC(40),SRUDM(30),MSOR(130),FMC(130),MQPA(40)

      IFLAG=0
      GO TO (1,2) I

C
C      CANNIBALIZATION ALLOWED
C
C      ARE LRUS AVAILABLE
C      IF YES, GET LRU
C      IF NO, USE FOR CANNIBALIZATION
C
1     J=INT(XX(55))
      DO 10 I=1,J
C      print *, 'Thisalloc1 j,qpa,atrib',j,qpa(i),atrib(1),atrib(2)
C      print *,atrib(3),atrib(4),atrib(5),atrib(6),atrib(7),atrib(8)
      IF (NNRSC(I).LT.QPA(I)) GO TO 30
10    CONTINUE
      DO 20 I=1,J
      LRUQPA=INT(QPA(I))
      CALL SEIZE(I,LRUQPA)
20    CONTINUE
      IFLAG=-1
C      print
*, 'thisalloc1a',j,qpa(i),atrib(1),atrib(2),atrib(3),atrib(4)
C      print *,atrib(5),atrib(6),atrib(7),atrib(8)
30    RETURN

C
C      CANNIBALIZATION NOT ALLOWED
C
C      ARE LRUS AVAILABLE ?
C      IF YES, GET LRU
C      IF NO, WAIT FOR PART
C
2     LRU1=INT(ATRIB(2))
      LRU2=INT(ATRIB(9))
C      print *, 'this is alloc2'
      IF (NNRSC(LRU1).LE.0) GO TO 40
      IF (LRU2.LE.0) GO TO 35
      IF (NNRSC(LRU2).LE.0) GO TO 40
      CALL SEIZE(LRU2,1)
      ATRIB(LRU2+10)=ATRIB(LRU2+10)+1
35    CALL SEIZE(LRU1,1)
      ATRIB(LRU1+10)=ATRIB(LRU1+10)+1
      IFLAG=-1

```



```
40  RETURN
    END

    FUNCTION USERF(I)

COMMON/SCOM1/ATTRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)

    REAL FLYHRS

    GO TO (1,2) I

1    USERF = AINT(TNOW/24)-AINT(ATTRIB(1)/24)
    RETURN

2    FLYHRS=24/(XX(7)*XX(70)*XX(71))
C    FLYHRS=23.98-(TNOW-XX(52))
C    IF (FLYHRS.GT.XX(4)) THEN
C        USERF=XX(4)
C        RETURN
C    ENDIF
    USERF=FLYHRS
    RETURN
    END
```

APPENDIX D

DATABASE MANAGEMENT SYSTEM CODE

This is the Dbase III+ code used to generate the data management system.

```

*.:*****
*
*.:
*.:      Program: BASEAID.PRG
*.:
*.:      System: disc-data-handler
*.:      Author: t.lewis
*.:      Copyright (c) 1994, t.lewis
*.:      Last modified: 07/16/95      4:42
*.:
*.:      Calls: BWRSKNAME.PRG
*.:              : BREVIEW.PRG
*.:              : BMTBD.PRG
*.:              : BDMDRATE.PRG
*.:              : BCHECK.PRG
*.:              : BBUILD.PRG
*.:              : BUTIL.PRG
*.:              : REPORT.PRG
*.:
*.:      Uses: WRSKNAME.DBF
*.:              : N&PDB.DBF
*.:              : WRSKBLNK.DBF
*.:              : &PDB.DBF
*.:
*.:      Indexes: &PDB
*.:
*.:      Documented: 07/16/95 at 04:44      SNAP! version
3.10a
*.:*****
*
```

```

CLOSE ALL
SET FUNCTION 10 TO CHR(23)
SET TALK OFF
SET BELL OFF
SET STATUS ON
SET ESCAPE ON
SET CONFIRM ON
SET COLOR TO W+/N,W+/B
PUBLIC PDB, PRNT
CLEAR
PRNT="Y"
@ 10, 8 SAY [ARE YOU USING A PRINTER ?];
    GET PRNT PICTURE "@!";
READ
PRNT=UPPER(PRNT)

TPDB = [      ]

IF LEN(TRIM(TPDB)) = 0
    TPDB = [      ]
    PDB = [      ]

```

```

CLEAR
DO WHILE .NOT. FILE(TPDB)
  @ 10, 5 SAY "Which database should I use? " GET PDB
  READ
  IF PDB = " "
    RETURN
  ENDIF
  PDB = UPPER(LTRIM(TRIM(PDB)))
  TPDB = [&pdb] + [DBF]
  WPDB=[N]+[&PDB]+[DBF]
  IF .NOT. FILE(TPDB)
    @ 12, 5 SAY "The file &tpdb does not exist."
    ANS = [Y]
    @ 14, 5 SAY "Shall I create it? " GET ANS PICTURE [L]
    READ
    ANS = ANS$[YyTt]
    IF ANS
      TPDB = [&pdb] + [DBF]
      NPDB = [&pdb] + [NTX]

      SET SAFE OFF
      SELE 9
      USE WRSKNAME
      COPY STRU TO N&PDB
      USE N&PDB
      APPE BLANK

      TBASE1=BASE1
      CLEAR
      @ 3, 6 SAY "PLEASE GIVE THE FOLLOWING KIT IDENTIFIERS

? "

      @ 5, 30 SAY "BASE NAME: "
      @ 5, 42 GET TBASE1
      @ 6, 32 SAY "KIT 1:"
      @ 6, 38 GET APP1
      @ 7, 32 SAY "KIT 2:"
      @ 7, 38 GET APP2
      @ 8, 32 SAY "KIT 3:"
      @ 8, 38 GET APP3
      @ 9, 32 SAY "KIT 4:"
      @ 9, 38 GET APP4
      @ 10, 32 SAY "KIT 5:"
      @ 10, 38 GET APP5
      @ 11, 32 SAY "KIT 6:"
      @ 11, 38 GET APP6
      @ 12, 32 SAY "KIT 7:"
      @ 12, 38 GET APP7
      @ 13, 32 SAY "KIT 8:"
      @ 13, 38 GET APP8
      @ 14, 32 SAY "KIT 9:"
      @ 14, 38 GET APP9
      @ 15, 32 SAY "KIT10:"
      @ 15, 38 GET APP10
      @ 16, 32 SAY "KIT11:"
      @ 16, 38 GET APP11
      @ 17, 32 SAY "KIT12:"

```

```

@ 17, 38 GET APP12
CLEAR TYPEAHEAD
READ
REPLACE BASE1 WITH TBASE1
REPLACE DFH WITH 0
REPLACE DFH30 WITH 0
REPLACE DSETUP WITH 0
REPLACE DFHSET WITH 0
REPLACE DFHD1 WITH 0
REPLACE DFHD2 WITH 0
REPLACE DFHD3 WITH 0
REPLACE DFHD4 WITH 0
REPLACE DFHD5 WITH 0
REPLACE DFHD6 WITH 0

SELE 1

IF .NOT. FILE(TPDB)
    USE WRSKBLNK
    COPY STRU TO &PDB
    USE &PDB
    INDEX ON NSN TO &PDB
ELSE
    IF .NOT. FILE(NPDB)
        USE &PDB
        INDEX ON NSN TO &PDB
    ENDIF
ENDIF
SET SAFE ON
ELSE
    CLEAR
    PDB=[          ]
    LOOP
ENDIF
ENDIF

CLEAR
IF .NOT. FILE(WPDB)
    @ 12, 5 SAY "The file for base and kit names doesn't exist."
    @ 14, 5 SAY "I will create it. "
    SELE 9
    SET SAFE OFF
    USE WRSKNAME
    COPY STRU TO N&PDB
    USE N&PDB
    APPE BLANK
    TBASE1=BASE1
    CLEAR
    @ 3, 6 SAY "PLEASE GIVE THE FOLLOWING KIT IDENTIFIERS ?"
    @ 5, 30 SAY "BASE NAME:"
    @ 5, 42 GET TBASE1
    @ 6, 32 SAY "KIT 1:"
    @ 6, 38 GET APP1
    @ 7, 32 SAY "KIT 2:"
    @ 7, 38 GET APP2
    @ 8, 32 SAY "KIT 3:"
    @ 8, 38 GET APP3

```

```

@ 9, 32 SAY "KIT 4:"
@ 9, 38 GET APP4
@ 10, 32 SAY "KIT 5:"
@ 10, 38 GET APP5
@ 11, 32 SAY "KIT 6:"
@ 11, 38 GET APP6
@ 12, 32 SAY "KIT 7:"
@ 12, 38 GET APP7
@ 13, 32 SAY "KIT 8:"
@ 13, 38 GET APP8
@ 14, 32 SAY "KIT 9:"
@ 14, 38 GET APP9
@ 15, 32 SAY "KIT10:"
@ 15, 38 GET APP10
@ 16, 32 SAY "KIT11:"
@ 16, 38 GET APP11
@ 17, 32 SAY "KIT12:"
@ 17, 38 GET APP12
CLEAR TYPEAHEAD
READ
REPLACE BASE1 WITH TBASE1
REPLACE DFH WITH 0
REPLACE DFH30 WITH 0
REPLACE DSETUP WITH 0
REPLACE DFHSET WITH 0
REPLACE DFHD1 WITH 0
REPLACE DFHD2 WITH 0
REPLACE DFHD3 WITH 0
REPLACE DFHD4 WITH 0
REPLACE DFHD5 WITH 0
REPLACE DFHD6 WITH 0

      SELE 1
    ENDIF
  ENDDO
ELSE
  PDB = UPPER(TPDB)
ENDIF

SET BELL OFF
USE &PDB
SET SAFE OFF
INDEX ON NSN TO &PDB
USE &PDB INDEX &PDB
SET DELETED OFF

SELE 9
USE N&PDB
IF RECCOUNT(<) < 1
  APPEND BLANK
  @ 10,15 SAY [WARNING: KIT SERIAL #s HAVE NOT BEEN INPUT.]
  @ 11,24 SAY [SEE OPTION # 1]
  SET CONFIRM OFF
  STORE ' ' TO WAIT_SUBST
  @ 23,0 SAY 'Press any key to continue...' GET WAIT_SUBST
  READ

```

```

      SET CONFIRM ON
ENDIF
SET SAFE ON
TBASE1=BASE1
TAPP1=APP1
TAPP2=APP2
TAPP3=APP3
TAPP4=APP4
TAPP5=APP5
TAPP6=APP6
TAPP7=APP7
TAPP8=APP8
TAPP9=APP9
TAPP10=APP10
TAPP11=APP11
TAPP12=APP12

SELE &PDB

DO WHILE .T.

    * ---Display menu options, centered on the screen.
    *   draw menu border and print heading
    CLEAR
    @ 2, 0 TO 18,79 DOUBLE
    @ 3,20 SAY [D I S S E R T A T I O N   D A T A   A I D]
    @ 4,1 TO 4,78 DOUBLE
    * ---display detail lines
    @ 7,19 SAY [1. EDIT KIT NAMES]
    @ 8,19 SAY [2. UTILITIES]
    @ 9,19 SAY [3. REVIEW WRSK DATA]
    @ 10,19 SAY [4. COMPUTE MTBDs]
    @ 11,19 SAY [5. COMPUTE DEMAND RATES]
    @ 12,19 SAY [6. CHECK FOR DATA ERRORS (INC. INDENTURE
RELATIONSHIPS)]
    @ 13,19 SAY [7. BUILD MODEL FILES]
    @ 14,19 SAY [8. REPORTS]
    @ 16,19 SAY [ 0. EXIT]
    STORE 0 TO SELECTNUM
    @ 18,33 SAY " select      "
    @ 18,42 GET SELECTNUM PICTURE "9" RANGE 0,8

    READ

    DO CASE
    CASE SELECTNUM = 0
        SET BELL ON
        SET TALK ON
        CLEAR ALL
        RETURN

    CASE SELECTNUM = 1
        DO BWRSKNAME

```

```

CASE SELECTNUM = 3
  * DO REVIEW WRSK DATA
  DO BREVVIEW

CASE SELECTNUM = 4
  * COMPUTE MTBDs
  DO BMTBD

CASE SELECTNUM = 5
  * DO COMPUTE DEMAND RATES
  DO BDMDRATE

CASE SELECTNUM = 6
  * DOES INDENTURE RELATIONSHIP CHECKS
  DO BCHECK

CASE SELECTNUM = 7
  * DO BUILD DMAS FILE
  DO BBUILD

CASE SELECTNUM = 2
  * FILE UTILITIES
  DO BUTIL

CASE SELECTNUM = 8
  DO REPORT

```

```

ENDCASE

```

```

ENDDO T
CLOSE ALL
RETURN

```

```

*: EOF: BASEAID.PRG

```

```

*: *****
*
*:
*:      Program: BBUILD.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/25/95      16:06
*:
*:      Called by: BASEAID.PRG
*:
*:      Calls: DMV4OUT.PRG
*:             : DMV6OUT.PRG

```



```

*:          : SIMOUT.PRG
*:          : SORTCOMB.PRG
*:
*:          Uses: TEMP.DBF
*:
*:          Indexes: TEMP.NDX
*:
*:          Documented: 07/16/95 at 04:44          SNAP! version
3.10a
*:*****
*
REINDEX
SET ECHO OFF
CLEAR
SET DELETED ON
FTXT=[      ]
TXT=[      ]
@15,5 SAY "GIVE NAME FOR DMAS INPUT FILE" GET TXT
@15,44 SAY ".TXT"
READ
IF TXT = " "
    RETURN
ENDIF
TXT=UPPER(TRIM(LTRIM(TXT)))
FTXT=[&TXT]+[.TXT]
SET ALTE TO &FTXT
SELE 9
APPNO=[      ]
CLEAR
?
?
?"THIS IS A LIST OF KIT NAMES"
?
LIST FIELDS
APP1,APP2,APP3,APP4,APP5,APP6,APP7,APP8,APP9,APP10,APP11,APP12 OFF

@12,5 SAY "GIVE THE APPROPRIATE KIT NAME." GET APPNO
READ
GO TOP
IF APPNO=" "
    SELE 1
    RETURN
ENDIF
NUM=1
DO WHILE NUM<13
    BNUM=TRIM(LTRIM(STR(NUM,2)))
    S=APP&BNUM
    IF APPNO=S
        APPNO=BNUM
        NUM=50
    ELSE
        NUM=NUM+1
    ENDIF
ENDDO
SELE 1
IF NUM < 30
    CLEAR

```

```

@7, 5 SAY "THIS KIT NAME "+APPNO+" DOES NOT EXIST IN MY DATABASE"
@8, 5 SAY "PRESS ANY KEY TO CONTINUE"
WAIT " "
RETURN
ENDIF
CHECK=1
DO WHILE CHECK<4
  CLEAR
  STORE " " TO BNAME
  @ 15,5 SAY;
    " PLEASE GIVE BASE NAME ";
  GET BNAME PICTURE "@!"
  READ
  IF BNAME="TEST"
    @15,5 SAY "TEST IS A DMAS RESTRICTED WORD. PLEASE CHOOSE
SOMETHING ELSE."
    @17,5 SAY "HIT ANY KEY TO CONTINUE"
    WAIT " "
  ELSE
    CHECK=5
  ENDIF
ENDDO
CLEAR
ANS=[N]
@ 15,5 SAY "DO YOU WANT TO USE AUTH QTY=999? (Y/N)" GET ANS
@ 18,5 SAY "NOTE: DMAS ROBUSTS TO AUTH QTYS"
READ
ANS=UPPER(ANS)
SET TALK OFF
SET SAFETY OFF
COPY TO TEMP
MESSAGE=1
SELE 3
USE TEMP
DELETE FOR APP&APPNO <=0
DELETE FOR REVIEWED<>"Y"
PACK
REPLACE ALL NSN WITH SUBSTR(NSN,1,13)
INDEX ON SUBSTR(NSN,5,9) TO TEMP
GO TOP
NSN2A=[ ]
NSN2=[ ]
DO WHILE .NOT. EOF()
  NSN1=SUBSTR(NSN,5,9)
  NSN1A=NSN
  IF NSN1=NSN2
    LINE1="WARNING: THERE ARE TWO RECORDS WITH THE SAME NIIN. THESE
COULD BE"
    LINE2="DUPLICATE NSNs. THE NSNs ARE "+NSN1A+" "+NSN2A
    ?LINE1
    ?LINE2
    ?" "
  ENDIF
  NSN2=NSN1
  NSN2A=NSN1A
  SKIP
ENDDO

```

```

INDEX ON NSN+UPPER(REVIEWED) TO TEMP
SET SAFETY ON
REPLACE ALL BASE WITH BNAME
IF ANS="Y"
    REPLACE ALL STK WITH 999
ENDIF

```

```

*****DO WHILE .T.

```

```

* ---Display menu options, centered on the screen.
*   draw menu border and print heading

```

```

CLEAR
@ 2, 0 TO 13,79 DOUBLE
@ 3,25 SAY [V E R S I O N   S E L E C T O R]
@ 4,1 TO 4,78 DOUBLE
* ---display detail lines
@ 7,29 SAY [1. Dyna-METRIC VER 4.4]
@ 8,29 SAY [2. Dyna-METRIC VER 6.0]
@ 9,29 SAY [3. RESEARCH MODEL]
@ 11, 29 SAY '0. EXIT'
STORE 0 TO SELECTNUM
@ 13,33 SAY " select      "
@ 13,42 GET SELECTNUM PICTURE "9" RANGE 0,3
READ

```

```

DO CASE
CASE SELECTNUM = 0
    USE
    SET PRINT OFF
    SET ALTE TO
    SET DELE OFF
    SELE 1

```

```

    RETURN

```

```

CASE SELECTNUM = 1
    * DO Dyna-METRIC VER 4.4
    DO DMV4OUT

```

```

CASE SELECTNUM = 2
    * DO Dyna-METRIC VER 6.0
    DO DMV6OUT

```

```

CASE SELECTNUM = 3
    * DO Simulation
    DO SIMOUT

```

```

ENDCASE

```

```

*****ENDDO T

```

```

SELE 2

```

```

CLEAR

```

```

?" THESE LRU HAVE SRUs INDENTURED TO THEM BUT ARE LISTED AS RR."

```

```

IF PRNT="Y"
  DISPLAY ALL LRU,SRU FOR ERROR2='Y' OFF TO PRINT
ELSE
  DISPLAY ALL LRU,SRU FOR ERROR2='Y' OFF
  @ 21,5 SAY "HIT ANY KEY TO CONTINUE"
  WAIT " "
ENDIF
CLEAR
?" THESE SRUs HAVE BEEN INCORRECTLY INDENTURED TO OTHER SRUs."
IF PRNT="Y"
  DISPLAY ALL LRU,SRU FOR ERROR2='L' OFF TO PRINT
ELSE
  DISPLAY ALL LRU,SRU FOR ERROR2='L' OFF
  @ 21,5 SAY "HIT ANY KEY TO CONTINUE"
  WAIT " "
ENDIF
SELE 3

IF PRNT="Y"
  SET PRINT ON
ENDIF
GO TOP
CLEAR
CNT = 4
DO WHILE .NOT. EOF()
  IF LRU_SRU="S"
    NSN1=SUBSTR(INDT_NSN1,1,13)
    NSN2=SUBSTR(INDT_NSN2,1,13)
    NSN3=SUBSTR(INDT_NSN3,1,13)
    NSN4=SUBSTR(INDT_NSN4,1,13)
    NSN5=SUBSTR(INDT_NSN5,1,13)
    NSN6=SUBSTR(INDT_NSN6,1,13)
    NSN7=SUBSTR(INDT_NSN7,1,13)
    NSN8=SUBSTR(INDT_NSN8,1,13)
    SRU=NSN
    SQPA=QPA
    QP1=QPL1
    QP2=QPL2
    QP3=QPL3
    QP4=QPL4
    QP5=QPL5
    QP6=QPL6
    QP7=QPL7
    QP8=QPL8
    SEEK TRIM(NSN1)
    LQP1=QPA
    TQPA= QP1 * LQP1
    IND='2'
    DO WHILE VAL(IND) < 9
      IF LEN(TRIM(NSN&IND))>0
        SEEK TRIM(NSN&IND)
        LQP&IND=QPA
        TQPA= TQPA+(QP&IND*LQP&IND)
      ENDIF
      IND=TRIM(LTRIM(STR(VAL(IND)+1)))
    ENDDO
    SEEK TRIM(SRU)
  
```

```

      IF TQPA<>SQPA
        MESSAGE=5
        IF PRNT<>"Y"
          @ CNT, 5 SAY "THE QPA AND QPLs FOR "+SRU+" DO NOT SUM
CORRECTLY"
          CNT = CNT + 1
        ELSE
          LINE1="THE QPA AND QPLs FOR "+SRU+" DO NOT SUM CORRECTLY"
          ?LINE1
        ENDIF
      ENDIF
    ENDIF
  SKIP
  IF (CNT>20 .OR. EOF()) .AND. CNT>4
    @ 21, 5 SAY "HIT ANY KEY TO CONTINUE"
    WAIT " "
    CLEAR
    CNT=4
  ENDIF
ENDDO

```

```

IF MESSAGE<5
  @ 13, 5 SAY "NO ERRORS FOUND"
  @ 15, 5 SAY "PLEASE PRESS ANY KEY TO CONTINUE"
  WAIT " "
ENDIF
*USE
IF PRNT="Y"
  EJECT
ENDIF
SELE 2
SET SAFE OFF
ZAP
SET SAFE ON
SET PRINT OFF
IF SELECTNUM=3
  SELE 3
  SET SAFE OFF
  INDEX ON NSN TO TEMP
  SET SAFE ON
  SELE 2
  DO SORTCOMB
ENDIF
SET ALTE TO
SET DELE OFF
SELE 1

```

```

*: EOF: BBUILD.PRG

```

```

*:*****
*
*:
*:      Program: BCHECK.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis

```

```

*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/24/95      17:21
*:
*:      Called by: BASEAID.PRG
*:
*:      Uses: TEMP.DBF
*:            : INDT.DBF
*:
*:      Indexes: TEMP.NDX
*:            : LRU.NDX
*:
*:      Documented: 07/16/95 at 04:44                      SNAP! version
3.10a
*:*****
*
SELE 9
APPNO=[      ]
CLEAR
?
?
?"THIS IS A LIST OF KIT NAMES"
?
LIST FIELDS
APP1,APP2,APP3,APP4,APP5,APP6,APP7,APP8,APP9,APP10,APP11,APP12 OFF

@12,5 SAY "GIVE THE APPROPRIATE KIT NAME." GET APPNO
READ
IF APPNO=" "
    RETURN
ENDIF
APPNO=(TRIM(LTRIM(APPNO)))

NUM=1
DO WHILE NUM<13
    BNUM=LTRIM(STR(NUM,2))
    IF APPNO=TRIM(APP&BNUM)
        APPNO=BNUM
        NUM=50
    ELSE
        NUM=NUM+1
    ENDIF
SELE 1
ENDDO
IF NUM < 30
    CLEAR
    @7, 5 SAY "THIS KIT NAME DOES NOT EXIST IN MY DATABASE"
    @8, 5 SAY "PRESS ANY KEY TO CONTINUE"
    WAIT " "
    RETURN
ENDIF
CLEAR
?" "
?"THE FOLLOWING CONDITIONS CAUSE FATAL ERRORS DURING DMAS PROCESSING"
IF PRNT="Y"
    @ 12,10 SAY [IF ERRORS ARE FOUND, THEY WILL BE PRINTED OUT.]
    SET PRINT ON
ENDIF

```

```

SET TALK OFF
SET SAFETY OFF
COPY TO TEMP
MESSAGE=1
SELE 3
USE TEMP
DELETE FOR APP&APPNO <=0
DELETE FOR REVIEWED<>"Y"
PACK
REPLACE ALL NSN WITH SUBSTR(NSN,1,13)
INDEX ON SUBSTR(NSN,5,9) TO TEMP
GO TOP
NSN2A=[           ]
NSN2=[           ]
DO WHILE .NOT. EOF()
    NSN1=SUBSTR(NSN,5,9)
    NSN1A=NSN
    IF NSN1=NSN2
        LINE1="WARNING: THERE ARE TWO RECORDS WITH THE SAME NIIN.  THESE
COULD BE"
        LINE2="DUPLICATE NSNs.  THE NSNs ARE "+NSN1A+"  "+NSN2A
        ?LINE1
        ?LINE2
        ?" "
    ENDIF
    NSN2=NSN1
    NSN2A=NSN1A
    SKIP
ENDDO
INDEX ON NSN+UPPER(REVIEWED) TO TEMP
SET SAFETY ON
GO TOP
SELECT 2
USE IND1
SET SAFETY OFF
ZAP
SET SAFETY ON
SELECT 3
NLRU=0
NSRU=0
MESSAGE=1
DO WHILE .NOT. EOF()
    IF .NOT. (LRU_SRU="S")
        NLRU=NLRU+1
        SKIP
        LOOP
    ENDIF
    NSRU=NSRU+1
    TSN=NSN
    LSN1=SUBSTR(INDT_NSN1,1,13)
    LSN2=SUBSTR(INDT_NSN2,1,13)
    LSN3=SUBSTR(INDT_NSN3,1,13)
    LSN4=SUBSTR(INDT_NSN4,1,13)
    LSN5=SUBSTR(INDT_NSN5,1,13)
    LSN6=SUBSTR(INDT_NSN6,1,13)
    LSN7=SUBSTR(INDT_NSN7,1,13)
    LSN8=SUBSTR(INDT_NSN8,1,13)

```

```

IND='2'
TQPL=QPA
SELECT INDT
APPEND BLANK
REPLACE SRU WITH TSN
REPLACE LRU WITH LSN1
DO WHILE VAL(IND) < 9
    IF LEN(TRIM(LSN&IND)) > 0
        APPEND BLANK
        REPLACE SRU WITH TSN
        REPLACE LRU WITH LSN&IND
    ENDIF
    IND=TRIM(LTRIM(STR(VAL(IND)+1)))
ENDDO
SELE 3
SKIP
ENDDO
SELE INDT
GO TOP
SET SAFE OFF
INDEX ON LRU TO LRU
SET SAFE ON

IF RECCOUNT () >0
    PLRU=[ ]
    STORE 0.00 TO TOTAL1,COST2
    DO WHILE .NOT. EOF()
        IF PLRU<>LRU
            IF TOTAL1>COST2
                LINE2="WARNING: THE FOLLOWING LRU IS CHEAPER THAN IT'S
COMBINED SRUs "+PLRU+" "+WUC2
                LINE3="lru cost"+STR(COST2,7)+" SRUs cost "+STR(TOTAL1,7)
                ?LINE2
                ?LINE3
            ENDIF
            STORE 0 TO TOTAL1
        ENDIF
        PLRU=LRU
        PSRU=TRIM(SRU)
        SELE 3
        IF PLRU=" "
            MESSAGE=5
            SELE 2
            REPLACE ERROR WITH "M"
            SELE 3
        ELSE
            SEEK TRIM(PLRU)
            IF .NOT. FOUND()
                MESSAGE=5
                SELE 2
                REPLACE ERROR WITH "D"
                SELE 3
            ELSE
                TMDR=DMD_RATE
                COST1=COST
                COST3=COST1*QPA
                TOTAL1=TOTAL1+COST3
            ENDIF
        ENDIF
    ENDWHILE

```



```

        IF RR_RRR<>1
            SELE 2
            REPLACE ERROR2 WITH "Y"
            MESSAGE=5
            SELE 3
        ENDIF
        IF LRU_SRU="S"
            SELE 2
            REPLACE ERROR2 WITH "L"
            MESSAGE=5
            SELE 3
        ENDIF
        SEEK PSRU
        SMDR=DMD_RATE
        COST2=COST
        WUC2=WUC
        IF COST1>COST
            LINE1=NSN1+" the preceding sru costs more than its lru.
"+NSN2
            ?LINE1
            ?COST1
            ?COST2
        ENDIF
        IF SMDR>TMDR
            MESSAGE=5
            SELE 2
            REPLACE LDR WITH TMDR,SDR WITH SMDR,ERROR WITH "W"
            SELE 3
        ENDIF
    ENDIF
    ENDIF
    SELE INDT
    SKIP
    ENDDO
ENDIF

SELE 2
ERR1=0
ERR2=0
ERR3=0
ERR4=0
ERR5=0
DO WHILE .NOT. EOF()
    IF ERROR<>" ".OR. ERROR2<>" "
        MESSAGE=5
        IF ERROR="M"
            ERR1=1
        ENDIF
        IF ERROR="D"
            ERR2=1
        ENDIF
        IF ERROR="W"
            ERR3=1
        ENDIF
        IF ERROR2="Y"
            ERR4=1
        ENDIF
    
```

```

        IF ERROR2="L"
            ERR5=1
        ENDIF
    ENDIF
    IF ERR1+ERR2+ERR3+EER4+ERR5>4
        EXIT
    ENDIF
    SKIP
ENDDO
GO TOP
IF ERR1>0
    ?" THESE SRUs ARE NOT INDENTURED TO AN LRU AND HAVE REVIEWED=Y."
    IF PRNT="Y"
        DISPLAY ALL SRU FOR ERROR='M' OFF TO PRINT
    ELSE
        DISPLAY ALL SRU FOR ERROR='M' OFF
        @ 21,5 SAY "HIT ANY KEY TO CONTINUE"
        WAIT " "
    ENDIF
ENDIF
CLEAR
IF ERR2>0
    GO TOP
    ?" THESE SRU's LRUS HAVE BEEN DELETED, NOT REVIEWED, "
    ?" OR DON'T EXIST IN THE DATABASE."
    IF PRNT="Y"
        DISPLAY ALL FIELDS SRU,LRU FOR ERROR='D' OFF TO PRINT
    ELSE
        DISPLAY ALL SRU,LRU FOR ERROR='D' OFF
        @ 21,5 SAY "HIT ANY KEY TO CONTINUE"
        WAIT " "
    ENDIF
ENDIF
GO TOP
CLEAR
IF ERR3>0
    ?" THESE SRUs DEMAND RATES ARE LARGER THAN THEIR PARENT LRU."
    ?" THIS SHOULD NOT HAPPEN."
    IF PRNT="Y"
        DISPLAY ALL SRU,SDR,LRU,LDR FOR ERROR='W' OFF TO PRINT
    ELSE
        DISPLAY ALL SRU,SDR,LRU,LDR FOR ERROR='W' OFF
        @ 21,5 SAY "HIT ANY KEY TO CONTINUE"
        WAIT " "
    ENDIF
ENDIF
GO TOP
CLEAR
IF ERR4>0
    ?" THESE LRU HAVE SRUs INDENTURED TO THEM BUT ARE LISTED AS RR."
    IF PRNT="Y"
        DISPLAY ALL LRU,SRU FOR ERROR2='Y' OFF TO PRINT
    ELSE
        DISPLAY ALL LRU,SRU FOR ERROR2='Y' OFF
        @ 21,5 SAY "HIT ANY KEY TO CONTINUE"
        WAIT " "
    ENDIF
ENDIF

```

```

ENDIF
GO TOP
CLEAR
IF ERR5>0
  ?" THESE SRUs HAVE BEEN INCORRECTLY INDENTURED TO OTHER SRUs."
  IF PRNT="Y"
    DISPLAY ALL LRU,SRU FOR ERROR2='L' OFF TO PRINT
  ELSE
    DISPLAY ALL LRU,SRU FOR ERROR2='L' OFF
    @ 21,5 SAY "HIT ANY KEY TO CONTINUE"
    WAIT " "
  ENDIF
ENDIF
ENDIF

```

***** MAY WANT TO INCLUDE QPA/QPL IN ERROR LIST

```

SELE 3
GO TOP
CLEAR
CNT = 4
DO WHILE .NOT. EOF()
  IF LRU_SRU="S"
    NSN1=SUBSTR(INDT_NSN1,1,13)
    NSN2=SUBSTR(INDT_NSN2,1,13)
    NSN3=SUBSTR(INDT_NSN3,1,13)
    NSN4=SUBSTR(INDT_NSN4,1,13)
    NSN5=SUBSTR(INDT_NSN5,1,13)
    NSN6=SUBSTR(INDT_NSN6,1,13)
    NSN7=SUBSTR(INDT_NSN7,1,13)
    NSN8=SUBSTR(INDT_NSN8,1,13)
    SRU=NSN
    SQPA=QPA
    QP1=QPL1
    QP2=QPL2
    QP3=QPL3
    QP4=QPL4
    QP5=QPL5
    QP6=QPL6
    QP7=QPL7
    QP8=QPL8
    SEEK TRIM(NSN1)
    LQP1=QPA
    TQPA= QP1 * LQP1
    IND='2'
    DO WHILE VAL(IND) < 9
      IF LEN(TRIM(NSN&IND))>0
        SEEK TRIM(NSN&IND)
        LQP&IND=QPA
        TQPA= TQPA+(QP&IND*LQP&IND)
      ENDIF
      IND=TRIM(LTRIM(STR(VAL(IND)+1)))
    ENDDO
    SEEK TRIM(SRU)
    IF TQPA<>SQPA
      MESSAGE=5
      IF PRNT<>"Y"

```

```

      @ CNT, 5 SAY "THE QPA AND QPLs FOR "+SRU+" DO NOT SUM
CORRECTLY"
      CNT = CNT + 1
    ELSE
      LINE1="THE QPA AND QPLs FOR "+SRU+" DO NOT SUM CORRECTLY"
      ?LINE1
    ENDIF
  ENDIF
ENDIF
SKIP
IF (CNT>20 .OR. EOF()) .AND. CNT>4
  @ 21, 5 SAY "HIT ANY KEY TO CONTINUE"
  WAIT " "
  CLEAR
  CNT=4
ENDIF
ENDDO

CLEAR
IF NLRU>750
  MESSAGE=5
  ?"WARNING: THERE ARE "+STR(NLRU)+" LRUs. DMAS ALLOWS A MAXIMUM OF
750 LRUs."
  ?" "
ENDIF
IF NSRU>620
  MESSAGE=5
  ?"WARNING: THERE ARE "+STR(NSRU)+" SRUs. DMAS ALLOWS A MAXIMUM OF
620 SRUs."
  ?" "
ENDIF
GO TOP
ERR1=0
ERR2=0
ERR3=0
DO WHILE .NOT. EOF()
  IF REVIEWED="Y" .AND. (COST<=0 .OR. DMD_RATE<=0 .OR. WADJ<=0)
    MESSAGE=5
    IF COST<=0
      ERR1=1
    ENDIF
    IF DMD_RATE<=0
      ERR2=1
    ENDIF
    IF WADJ<=0
      ERR3=1
    ENDIF
  ENDIF
  IF ERR1+ERR2+ERR3>2
    EXIT
  ENDIF
SKIP
ENDDO

GO TOP
IF ERR1=1
  ?" "

```

```

?" THESE NSN HAVE A 0 DEMAND RATE AND HAVE REVIEWED = Y."
IF PRNT="Y"
    DISPLAY ALL NSN FOR DMD_RATE<=0 .AND. REVIEWED="Y" OFF TO PRINT
ELSE
    DISPLAY ALL NSN FOR DMD_RATE<=0 .AND. REVIEWED="Y" OFF
    @ 21,5 SAY "HIT ANY KEY TO CONTINUE"
    WAIT " "
ENDIF
ENDIF
CLEAR
IF ERR2=1
    ?" "
    ?" THESE NSN HAVE A 0 COST AND REVIEWED = Y."
    IF PRNT="Y"
        DISPLAY ALL NSN FOR COST<=0 .AND. REVIEWED='Y'OFF TO PRINT
    ELSE
        DISPLAY ALL NSN FOR COST<=0 .AND. REVIEWED='Y'OFF
        @ 21,5 SAY "HIT ANY KEY TO CONTINUE"
        WAIT " "
    ENDIF
ENDIF
CLEAR
IF ERR3=1
    ?" "
    ?" THESE NSN HAVE A 0 WARTIME ADJUSTMENT FACTOR AND HAVE REVIEWED =
Y."
    ?" THE STANDARD DEFAULT VALUE IS 1.0"
    IF PRNT="Y"
        DISPLAY ALL NSN FOR WADJ<=0 .AND. REVIEWED="Y" OFF TO PRINT
    ELSE
        DISPLAY ALL NSN FOR WADJ<=0 .AND. REVIEWED="Y" OFF
        @ 21,5 SAY "HIT ANY KEY TO CONTINUE"
        WAIT " "
    ENDIF
ENDIF
ENDIF

USE
CLEAR
?" "
?" "
?" END OF CHECK"
IF MESSAGE<5
    @ 13, 5 SAY "NO ERRORS FOUND"
    @ 15, 5 SAY "PLEASE PRESS ANY KEY TO CONTINUE"
    WAIT " "
ENDIF
SELE 2
SET SAFE OFF
ZAP
SET SAFE ON
IF PRNT="Y"
    EJECT
    SET PRINT OFF
ENDIF
SET DELE OFF
SELE 1
GO TOP

```

*: EOF: BCHECK.PRG

```

*:*****
*
*:
*:      Program: BDMDRATE.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/24/95      17:21
*:
*:      Called by: BASEAID.PRG
*:
*:      Documented: 07/16/95 at 04:44                      SNAP! version
3.10a
*:*****
*
SET TALK OFF
CLEAR
ANS=[ ]
@10,5 SAY [DOES THIS KIT HAVE RRR NOPs ? (Y/N)] GET ANS PICTURE [N]
READ
MESSAGE=1
SELE N&PDB
ANS=UPPER(ANS)
IF ANS<>"N"
    FH      = DFH
    FHSET = DFHSET
    FH30    = DFH30
    SETUP   = DSETUP
    FHD1    = DFHD1
    FHD2    = DFHD2
    FHD3    = DFHD3
    FHD4    = DFHD4
    FHD5    = DFHD5
    FHD6    = DFHD6
    GOODSCEN = .F.
    DO WHILE .NOT. GOODSCEN
        CLEAR
        @ 1,5 SAY [NOP DEMAND RATE COMPUTATIONS]
        @ 3,5 SAY [What are the total flying hours for this kit? ]
GET FH      PICT [99999.99]
        @ 5,5 SAY [          How many flying hours on the last day? ]
GET FH30    PICT [9999.99]
        @ 7,5 SAY [          How many days of setup are there?      ]
GET SETUP   PICT [99]
        @ 9,5 SAY [          How many flying hours during setup?    ]
GET FHSET   PICT [9999.99]
        STUP = 0
        @ 10,0 SAY [ ]
        DO WHILE STUP < 6
            STUP = STUP + 1
            ST = LTRIM(TRIM(STR(STUP,2,0)))
            STDAY = STR(SETUP + VAL(ST),3,0)

```

```

      @ ROW()+1,16 SAY [How many flying hours on day ]+STDAY+[ ?
] + STR(FHD&ST,7,2)
      ENDDO
      READ

      @ 3,5 SAY [What are the total flying hours for this kit? ]
+ STR(FH,8,2)
      @ 5,5 SAY [          How many flying hours on the last day? ] +
STR(FH30,7,2)
      @ 7,5 SAY [          How many days of setup are there?      ]
+ STR(SETUP,2,0)
      @ 9,5 SAY [          How many flying hours during setup?    ]
+ STR(FHSET,7,2)
      STUP = 0
      @ 10,0 SAY [ ]
      DO WHILE STUP < 6
        STUP = STUP + 1
        ST = LTRIM(TRIM(STR(STUP,2,0)))
        STDAY = STR(SETUP + VAL(ST),3,0)
        @ ROW()+1,16 SAY [How many flying hours on day ]+STDAY+[ ? ]
GET FHD&ST PICT [9999.99]
      ENDDO
      READ

      @ 23,10 SAY "Is this scenario correct? (Y/N) " GET GOODSCEN
PICT [Y]
      READ
      ENDDO

      REPLACE DFH      WITH FH
      REPLACE DFHSET   WITH FHSET
      REPLACE DFH30    WITH FH30
      REPLACE DSETUP   WITH SETUP
      REPLACE DFHD1    WITH FHD1
      REPLACE DFHD2    WITH FHD2
      REPLACE DFHD3    WITH FHD3
      REPLACE DFHD4    WITH FHD4
      REPLACE DFHD5    WITH FHD5
      REPLACE DFHD6    WITH FHD6

      GO TOP

ELSE
  CLEAR
  FH = DFH
  @ 10,5 SAY [NOP DEMAND RATE COMPUTATIONS]
  @ 12,5 SAY [What are the total flying hours for this kit? ] GET
FH PICT [99999.99]
  READ
  REPLACE DFH      WITH FH
ENDIF

IF FH<=0
  CLEAR
  @ 13,3 SAY [NO TOTAL FLYING HOURS WERE GIVEN, THEREFORE NO DEMAND
RATES WILL BE COMPUTED]

```

```

@ 20,10 SAY [PRESS ANY KEY TO CONTINUE]
WAIT ""
RETURN
ENDIF

SELE &PDB
GO TOP
CLEAR
@ 20,5 SAY "NOTE: COMPUTED DEMAND RATES WILL INCLUDE WARTIME AJUSTMENT
FACTORS"
CLEAR
@ 3, 5 SAY "NO DEMAND RATE WILL BE COMPUTED FOR THE FOLLOWING NSNs"
CNT = 4
DO WHILE .NOT. EOF()
  IF CNT > 20
    @ 21,5 SAY "PRESS ANY KEY TO CONTINUE"
    WAIT " "
    CLEAR
    CNT = 4
  ENDIF

  IF COMP='NOP'.OR. COMP='ADJ'
    NOP=(STK+((9/8)-((3/2)*SQRT(STK+(9/16))))))
    IF RR_RRR > 0
      IF BRCT > 0
        IF ANS<>"Y"
          MESSAGE=5
          @ CNT, 5 SAY [WARNING: ]+NSN+[ IS A RRR NOP. USER
STATED THERE WERE NONE.]
          CNT=CNT+1
          SKIP
          LOOP
        ENDIF
        STUP = 0
        FHBEG = FHSET
        DO WHILE STUP < BRCT
          STUP = STUP + 1
          ST = LTRIM(TRIM(STR(STUP,2,0)))
          FHBEG = FHBEG + FHD&ST
        ENDDO
        FBEG = FHBEG * QPA
        FEND = QPA * FH * NRTS_RT + FH30 * (1 - NRTS_RT) * BRCT
        NOPRRR=NOP/MAX(FBEG,FEND)
        REPLACE DMD_RATE WITH NOPRRR
      ELSE
        MESSAGE=5
        @ CNT, 5 SAY [The following NOP RRR nsn ]+NSN+[ has no
BRCT.]
        CNT=CNT+1
      ENDIF
    ELSE
      NOPRR=NOP/(FH * QPA)
      REPLACE DMD_RATE WITH NOPRR
    ENDIF
    SKIP
    LOOP
  ENDIF

```



```

IF .NOT.(MTBD>0)
  MESSAGE=5
  @ CNT, 5 SAY NSN+" NO MTBD"
  CNT=CNT+1
  SKIP
  LOOP
ENDIF
DR=(1/MTBD)
REPLACE DMD_RATE WITH DR
SKIP
ENDDO
IF MESSAGE<5
  @ 15,5 SAY "NO ERRORS FOUND"
ENDIF
@ 21,5 SAY "PRESS ANY KEY TO CONTINUE"
WAIT " "

```

*: EOF: BDMDRATE.PRG

```

*:*****
*
*:
*:      Program: BMTBD.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/24/95      17:21
*:
*:      Called by: BASEAID.PRG
*:
*:      Uses: FLYHRS.DBF
*:
*:      Indexes: FLYHRS.NDX
*:
*:      Documented: 07/16/95 at 04:44
*:
3.10a      SNAP! version
*:*****
*
SET SAFE OFF
SELE 2
USE FLYHRS
INDEX ON HOURS TO FLYHRS
SET SAFETY ON
CLEAR
@ 3, 5 SAY "THE FOLLOWING FLYING HOUR CODES EXIST IN THE DATA BASE"
@ 4, 5 SAY "IF THIS SCREEN IS BLANK NO CODES CURRENTLY EXIST"
DISPLAY ALL HOURS,FLYHR
@20, 5 SAY "PRESS ANY KEY TO CONTINUE"
WAIT " "

```

```

FQUIT="F"
DO WHILE FQUIT="F"
  CLEAR
  @ 10, 0
  STORE " " TO FLAG
  @10,2 SAY;
  " FLYING HOURS CODE TO (1)DEL / (2)ADD / ( )EDIT ";
  GET FLAG PICTURE "@!"
  @15,3 SAY "IF YOU ARE READY TO COMPUTE MTBDs OR WANT TO ESCAPE HIT
RETURN"
  READ
  IF FLAG <> " "

  DO CASE

  CASE FLAG = "1"

    CLEAR
    @ 10, 0
    STORE " " TO FLAG
    @10,2 SAY;
    " CODE TO DELETE ";
    GET FLAG PICTURE "@!"
    READ
    SEEK TRIM(FLAG)
    IF .NOT. FOUND ()
      @15,2 SAY;
      " CODE NOT FOUND "
    LOOP
  ENDIF

  STORE " " TO MCONFIRM
  @ 10,0
  @10,10 SAY "ARE YOU SURE YOU WANT TO DELETE THIS CODE ?
(Y/N) ";
  GET MCONFIRM PICTURE "@!"
  READ
  IF .NOT. (MCONFIRM="N")
    DELETE
    PACK
    @ 23,15 SAY TRIM(FLAG) +;
    " HAS BEEN DELETED -- PRESS ANY KEY TO CONTINUE"
    WAIT " "
  ENDIF
  LOOP

  CASE FLAG = "2"

  DO WHILE .NOT. (FLAG=" ")
    CLEAR
    STORE " " TO FLAG
    @ 10, 0
    @ 10,10 SAY "Enter the CODE for the new entry ";
    GET FLAG PICTURE "@!"
    READ

```

```

        IF FLAG = " "
            LOOP
        ENDIF
        SEEK FLAG
        IF FOUND ()
            @ 10,0
            @ 10,10 SAY TRIM (FLAG) +;
                " is already on file -- Press any key to contine"
            WAIT ""
        ELSE
            SET TALK OFF
            APPEND BLANK
            REPLACE HOURS WITH FLAG
        ENDIF
        CLEAR
        @ 7, 3 SAY "HOURS CODE:"
        @ 7, 15 GET HOURS
        @ 7, 22 SAY "TOTAL FLYING HOURS:"
        @ 7, 42 GET FLYHR
        READ
    ENDDO
    LOOP

    OTHERWISE
        SET TALK OFF
        SEEK TRIM(FLAG)
        IF .NOT. FOUND ()
            CLEAR
            STORE " " TO ANSW
            @ 10,15 SAY TRIM(FLAG) +;
                " NOT FOUND"
            @ 15,15 SAY "DO YOU WANT TO ADD THIS CODE ? (Y/N)";
                GET ANSW PICTURE "@!"
            READ
            IF ANSW = "N"
                LOOP
            ENDIF
            APPEND BLANK
            REPLACE HOURS WITH FLAG
        ENDIF
        CLEAR
        @ 7, 3 SAY "HOURS CODE:"
        @ 7, 15 GET HOURS
        @ 7, 22 SAY "TOTAL FLYING HOURS:"
        @ 7, 42 GET FLYHR
        READ
    ENDCASE
    ELSE
        FQUIT='T'
    ENDIF
    ENDDO
    SET SAFE OFF
    PACK
    SET SAFE ON
    CLEAR
    STORE " " TO ANSW
    @15,10 SAY "DO YOU WANT TO COMPUTE MTBDs (Y/N)?";

```

```

      GET ANSW PICTURE "@"
READ
IF .NOT. (ANSW='Y')
      SELE &PDB
      RETURN
ENDIF

SELE &PDB
GO TOP
CLEAR
@ 3,5 SAY "THE FOLLOWING NSN HAVE NO FLYING HOURS FLAG"
@ 4,5 SAY "THEREFORE NO MTBD WILL BE COMPUTED"
CNT = 5
MESSAGE=1
DO WHILE .NOT. EOF()
      IF HOURS=' '
            IF CNT > 20
                  @ 21, 5 SAY "HIT ANY KEY TO CONTINUE"
                  WAIT " "
                  CLEAR
                  CNT = 4
            ENDIF
            @ CNT, 5 SAY NSN
            CNT = CNT + 1
            MESSAGE=5
            SKIP
            LOOP
      ENDIF
      CODE=HOURS
      SELE FLYHRS
      SEEK CODE
      SELE &PDB
      DMDS=BASE1DMD
      FH=FLYHRS->FLYHR
      REPLACE TOTDMDS WITH DMDS
      REPLACE TOTFLYHRS WITH FH
      IF DMDS>0
            MTBD=(FH*QPA)/DMDS
            REPLACE MTBD WITH M->MTBD
      ENDIF
      SKIP
ENDDO
IF MESSAGE<5
      @ 15, 5 SAY "NO ERRORS FOUND"
ENDIF
@ 21, 5 SAY "HIT ANY KEY TO CONTINUE"
WAIT " "
SET SAFE ON

```

```

*: EOF: BMTBD.PRG

```

```

*:*****
*
*:
*:      Program: BREVIEW.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 07/16/95      4:35
*:
*:      Called by: BASEAID.PRG
*:
*:      Documented: 07/16/95 at 04:44      SNAP! version
3.10a
*:*****
*

```

```

GO TOP
SET DELETED OFF
NSN2=NSN
CLEAR
STORE "N" TO ANS
@ 10, 10 SAY "DO YOU WANT TO EDIT JUST DEMANDS ?";
  GET ANS PICTURE "@!"
READ
IF ANS="Y"
  DO WHILE .T.
    CLEAR
    STORE " " TO NSN1
    @10,2 SAY;
      "NSN TO MODIFY/INPUT DEMANDS ";
      GET NSN1 PICTURE "@!"
    @11,7 SAY [(P)REVIOUS / (N)EXT]
    READ
    CLEAR
    IF NSN1= " "
      RETURN
    ENDIF

    IF NSN1="P "
      SEEK TRIM(NSN2)
      IF .NOT. FOUND()
        @ 10, 5 SAY "PREVIOUS STOCK NUMBER WAS NOT ADDED. PLEASE
HIT ANY KEY TO CONTINUE."
        WAIT " "
        LOOP
      ENDIF
      NSN1=NSN
      NSN2=NSN1
      CMTBD= INT(CMDMTBD)
      CLEAR
      @ 1, 0 SAY "NSN:"
      @ 1, 4 GET NSN
      @ 1, 21 SAY "WUC:"
      @ 1, 25 SAY WUC
      @ 1, 31 SAY "NOUN:"
      @ 1, 36 SAY NOUN

```

```

@ 1, 62 SAY "COST:"
@ 1, 67 SAY COST
@ 3, 0 SAY "COMP:"
@ 3, 5 SAY COMP
@ 3, 9 SAY "STOCK:"
@ 3, 15 SAY STK
@ 3, 19 SAY "QPA:"
@ 3, 24 SAY QPA
@ 4, 15 SAY "MIN QPA:"
@ 4, 24 SAY MQPA
@ 3, 27 SAY "RCT:"
@ 3, 32 SAY BRCT
@ 3, 39 SAY "NRTS:"
@ 3, 45 SAY NRTS_RT
@ 3, 51 SAY "DEPOT:"
@ 3, 57 SAY DEPOT
@ 5, 45 SAY "WADJ:"
@ 5, 51 SAY WADJ PICTURE "99.99"
@ 3, 60 SAY "DMD RATE:"
@ 3, 70 SAY DMD_RATE
@ 4, 60 SAY "DEMANDS:"
@ 4, 69 GET BASE1DMD
@ 5, 63 SAY "MTBD:"
@ 5, 69 GET MTBD
@ 6, 60 SAY "FH CODE:"
@ 6, 69 GET HOURS
@ 5, 0 SAY "PERCENT APPLICATION:"
IF LEN(TRIM(TAPP1))>0
  @ 6, 0 SAY TAPP1+[:]
  @ 6, 6 SAY APP1
ENDIF
IF LEN(TRIM(TAPP2))>0
  @ 6, 13 SAY TAPP2+[:]
  @ 6, 19 SAY APP2
ENDIF
IF LEN(TRIM(TAPP3))>0
  @ 6, 26 SAY TAPP3+[:]
  @ 6, 32 SAY APP3
ENDIF
IF LEN(TRIM(TAPP4))>0
  @ 7, 0 SAY TAPP4+[:]
  @ 7, 6 SAY APP4
ENDIF
IF LEN(TRIM(TAPP5))>0
  @ 7, 13 SAY TAPP5+[:]
  @ 7, 19 SAY APP5
ENDIF
IF LEN(TRIM(TAPP6))>0
  @ 7, 26 SAY TAPP6+[:]
  @ 7, 32 SAY APP6
ENDIF
IF LEN(TRIM(TAPP7))>0
  @ 8, 0 SAY TAPP7+[:]
  @ 8, 6 SAY APP7
ENDIF
IF LEN(TRIM(TAPP8))>0
  @ 8, 13 SAY TAPP8+[:]

```

```

      @ 8, 19 SAY APP8
ENDIF
IF LEN(TRIM(TAPP9))>0
  @ 8, 26 SAY TAPP9+[:]
  @ 8, 32 SAY APP9
ENDIF
IF LEN(TRIM(TAPP10))>0
  @ 9, 0 SAY TAPP10+[:]
  @ 9, 6 SAY APP10
ENDIF
IF LEN(TRIM(TAPP11))>0
  @ 9, 13 SAY TAPP11+[:]
  @ 9, 19 SAY APP11
ENDIF
IF LEN(TRIM(TAPP12))>0
  @ 9, 26 SAY TAPP12+[:]
  @ 9, 32 SAY APP12
ENDIF
@ 11, 10 SAY "LRU/SRU:"
@ 11, 19 SAY LRU_SRU
@ 11, 23 SAY "MAINT (0-RR,1-RRR):"
@ 11, 43 SAY RR_RRR
@ 16, 0 SAY "CURRENTLY DELETED:"
@ 16, 18 SAY DEL
@ 16, 21 SAY "REVIEW:"
@ 16, 29 GET REVIEWED
@ 17, 0 SAY "REMARKS:"
@ 18, 0 GET REMARK
@ 19, 0 GET REMARKS
READ
REPLACE CMDMTBD WITH CMTBD
IF DEL="Y"
  DELETE
ELSE
  RECALL
ENDIF
LOOP
ENDIF

IF NSN1="N "
  IF EOF()
    CLEAR
    @10,5 SAY [THIS IS THE END OF THE FILE]
    @12,5 SAY [PLEASE STRIKE ANY KEY TO CONTINUE]
    WAIT " "
    SKIP -1
    NSN1=NSN
    NSN2=NSN1
    LOOP
  ENDIF
  SKIP
  IF EOF()
    CLEAR
    @10,5 SAY [THIS IS THE END OF THE FILE]
    @12,5 SAY [PLEASE STRIKE ANY KEY TO CONTINUE]
    WAIT " "

```

```

      SKIP -1
      NSN1=NSN
      NSN2=NSN1
      LOOP
ENDIF
NSN1=NSN
NSN2=NSN1
CMTBD= INT (CMDMTBD)
CLEAR
@ 1, 0 SAY "NSN:"
@ 1, 4 GET NSN
@ 1, 21 SAY "WUC:"
@ 1, 25 SAY WUC
@ 1, 31 SAY "NOUN:"
@ 1, 36 SAY NOUN
@ 1, 62 SAY "COST:"
@ 1, 67 SAY COST
@ 3, 0 SAY "COMP:"
@ 3, 5 SAY COMP
@ 3, 9 SAY "STOCK:"
@ 3, 15 SAY STK
@ 3, 19 SAY "QPA:"
@ 3, 24 SAY QPA
@ 4, 15 SAY "MIN QPA:"
@ 4, 24 SAY MQPA
@ 3, 27 SAY "RCT:"
@ 3, 32 SAY BRCT
@ 3, 39 SAY "NRTS:"
@ 3, 45 SAY NRTS_RT
@ 3, 51 SAY "DEPOT:"
@ 3, 57 SAY DEPOT
@ 5, 45 SAY "WADJ:"
@ 5, 51 SAY WADJ PICTURE "99.99"
@ 3, 60 SAY "DMD RATE:"
@ 3, 70 SAY DMD_RATE
@ 4, 60 SAY "DEMANDS:"
@ 4, 69 GET BASE1DMD
@ 5, 63 SAY "MTBD:"
@ 5, 69 GET MTBD
@ 6, 60 SAY "FH CODE:"
@ 6, 69 GET HOURS
@ 5, 0 SAY "PERCENT APPLICATION:"
IF LEN(TRIM(TAPP1))>0
  @ 6, 0 SAY TAPP1+[:]
  @ 6, 6 SAY APP1
ENDIF
IF LEN(TRIM(TAPP2))>0
  @ 6, 13 SAY TAPP2+[:]
  @ 6, 19 SAY APP2
ENDIF
IF LEN(TRIM(TAPP3))>0
  @ 6, 26 SAY TAPP3+[:]
  @ 6, 32 SAY APP3
ENDIF
IF LEN(TRIM(TAPP4))>0
  @ 7, 0 SAY TAPP4+[:]
  @ 7, 6 SAY APP4

```



```

ENDIF
IF LEN(TRIM(TAPP5))>0
  @ 7, 13 SAY TAPP5+[:]
  @ 7, 19 SAY APP5
ENDIF
IF LEN(TRIM(TAPP6))>0
  @ 7, 26 SAY TAPP6+[:]
  @ 7, 32 SAY APP6
ENDIF
IF LEN(TRIM(TAPP7))>0
  @ 8, 0 SAY TAPP7+[:]
  @ 8, 6 SAY APP7
ENDIF
IF LEN(TRIM(TAPP8))>0
  @ 8, 13 SAY TAPP8+[:]
  @ 8, 19 SAY APP8
ENDIF
IF LEN(TRIM(TAPP9))>0
  @ 8, 26 SAY TAPP9+[:]
  @ 8, 32 SAY APP9
ENDIF
IF LEN(TRIM(TAPP10))>0
  @ 9, 0 SAY TAPP10+[:]
  @ 9, 6 SAY APP10
ENDIF
IF LEN(TRIM(TAPP11))>0
  @ 9, 13 SAY TAPP11+[:]
  @ 9, 19 SAY APP11
ENDIF
IF LEN(TRIM(TAPP12))>0
  @ 9, 26 SAY TAPP12+[:]
  @ 9, 32 SAY APP12
ENDIF
@ 11, 10 SAY "LRU/SRU:"
@ 11, 19 SAY LRU_SRU
@ 11, 23 SAY "MAINT (0-RR,1-RRR):"
@ 11, 43 SAY RR_RRR
@ 16, 0 SAY "CURRENTY DELETED:"
@ 16, 18 SAY DEL
@ 16, 21 SAY "REVIEW:"
@ 16, 29 GET REVIEWED
@ 17, 0 SAY "REMARKS:"
@ 18, 0 GET REMARK
@ 19, 0 GET REMARKS
READ
IF DEL="Y"
  DELETE
ELSE
  RECALL
ENDIF
LOOP
ENDIF

SET TALK OFF
SEEK TRIM(NSN1)
IF .NOT. FOUND()
  @10,0

```

```

@10,15 SAY TRIM(NSN1)+;
      " NOT FOUND -- PRESS ANY KEY TO CONTINUE"
WAIT""
@10,0
STORE " " TO RESPONSE
@10,2 SAY;
      "do you want to add this (Y/N)?";
      GET RESPONSE PICTURE "@!"
READ
IF RESPONSE= "N"
      LOOP
ENDIF
IF RESPONSE="Y"
      SET TALK OFF
      APPEND BLANK
      REPLACE NSN WITH NSN1,QPA WITH 1,MQPA WITH 1,CANN_IND WITH
0
      REPLACE LRU_SRU WITH "L",NRTS_RT WITH 1.00,BRCT WITH 2.00
      REPLACE RR_RRR WITH 0,APPL WITH 1.00,WADJ WITH 1.0
      REPLACE REVIEWED WITH "N"
      REPLACE DEL WITH "N"
      IND='1'
      DO WHILE VAL(IND)<21
            REPLACE APP&IND WITH 1
            IND=TRIM(LTRIM(STR(VAL(IND)+1)))
      ENDDO
      CLEAR
      SET COLOR TO G+/N,W+/B
      @ 1, 0 SAY "NSN:"
      @ 1, 62 SAY "COST:"
      @ 3, 0 SAY "COMP:"
      @ 3, 9 SAY "STOCK:"
      @ 3, 19 SAY "QPA:"
      @ 4, 15 SAY "MIN QPA:"
      @ 3, 27 SAY "RCT:"
      @ 3, 39 SAY "NRTS:"
      @ 3, 60 SAY "DMD RATE:"
      @ 5, 45 SAY "WADJ:"
      @ 11, 10 SAY "LRU/SRU:"
      @ 11, 23 SAY "MAINT (0-RR,1-RRR):"
      SET COLOR TO R+/N
      @ 1, 4 GET NSN
      @ 1, 21 SAY "WUC:"
      @ 1, 25 GET WUC
      @ 1, 31 SAY "NOUN:"
      @ 1, 36 GET NOUN
      @ 1, 67 GET COST
      @ 3, 5 GET COMP
      @ 3, 15 GET STK
      @ 3, 24 GET QPA
      @ 4, 24 GET MQPA
      @ 3, 32 GET BRCT
      @ 3, 45 GET NRTS_RT
      @ 3, 51 SAY "DEPOT:"
      @ 3, 57 GET DEPOT
      @ 5, 51 GET WADJ PICTURE "99.99"
      @ 3, 70 GET DMD_RATE

```

```

@ 4, 60 SAY "DEMANDS:"
@ 4, 69 GET BASE1DMD
@ 5, 63 SAY "MTBD:"
@ 5, 69 GET MTBD
@ 6, 60 SAY "FH CODE:"
@ 6, 69 GET HOURS
SET COLOR TO G+/N,W+/B
@ 5, 0 SAY "PERCENT APPLICATION:"
IF LEN(TRIM(TAPP1))>0
  @ 6, 0 SAY TAPP1+[:]
  @ 6, 6 GET APP1
ENDIF
IF LEN(TRIM(TAPP2))>0
  @ 6, 13 SAY TAPP2+[:]
  @ 6, 19 GET APP2
ENDIF
IF LEN(TRIM(TAPP3))>0
  @ 6, 26 SAY TAPP3+[:]
  @ 6, 32 GET APP3
ENDIF
IF LEN(TRIM(TAPP4))>0
  @ 7, 0 SAY TAPP4+[:]
  @ 7, 6 GET APP4
ENDIF
IF LEN(TRIM(TAPP5))>0
  @ 7, 13 SAY TAPP5+[:]
  @ 7, 19 GET APP5
ENDIF
IF LEN(TRIM(TAPP6))>0
  @ 7, 26 SAY TAPP6+[:]
  @ 7, 32 GET APP6
ENDIF
IF LEN(TRIM(TAPP7))>0
  @ 8, 0 SAY TAPP7+[:]
  @ 8, 6 GET APP7
ENDIF
IF LEN(TRIM(TAPP8))>0
  @ 8, 13 SAY TAPP8+[:]
  @ 8, 19 GET APP8
ENDIF
IF LEN(TRIM(TAPP9))>0
  @ 8, 26 SAY TAPP9+[:]
  @ 8, 32 GET APP9
ENDIF
IF LEN(TRIM(TAPP10))>0
  @ 9, 0 SAY TAPP10+[:]
  @ 9, 6 GET APP10
ENDIF
IF LEN(TRIM(TAPP11))>0
  @ 9, 13 SAY TAPP11+[:]
  @ 9, 19 GET APP11
ENDIF
IF LEN(TRIM(TAPP12))>0
  @ 9, 26 SAY TAPP12+[:]
  @ 9, 32 GET APP12
ENDIF
SET COLOR TO W+/N,W+/B

```

```

@ 11, 19  GET  LRU_SRU
@ 11, 43  GET  RR_RRR
@ 13, 10  SAY [ IF PART IS SRU, INDENTURES ARE SHOWN ON
FOLLOWING SCREEN]
@ 15,  0  SAY "CURRENTY DELETED:"
@ 15, 18  GET  DEL
@ 15, 21  SAY "REVIEW:"
@ 15, 29  GET  REVIEWED
@ 17,  0  SAY "REMARKS:"
@ 18,  0  GET  REMARK
@ 19,  0  GET  REMARKS
CLEAR TYPEAHEAD
READ
IF LRU_SRU="S"
  CLEAR
  @ 4,  0  SAY "1. SRU INDENTURE TO:"
  @ 4, 21  GET  INDT_NSN1
  @ 4, 38  SAY "QPL:"
  @ 4, 43  GET  QPL1
  @ 6,  0  SAY "2. SRU INDENTURE TO:"
  @ 6, 21  GET  INDT_NSN2
  @ 6, 38  SAY "QPL:"
  @ 6, 43  GET  QPL2
  @ 8,  0  SAY "3. SRU INDENTURE TO:"
  @ 8, 21  GET  INDT_NSN3
  @ 8, 38  SAY "QPL:"
  @ 8, 43  GET  QPL3
  @ 10,  0  SAY "4. SRU INDENTURE TO:"
  @ 10, 21  GET  INDT_NSN4
  @ 10, 38  SAY "QPL:"
  @ 10, 43  GET  QPL4
  @ 12,  0  SAY "5. SRU INDENTURE TO:"
  @ 12, 21  GET  INDT_NSN5
  @ 12, 38  SAY "QPL:"
  @ 12, 43  GET  QPL5
  @ 14,  0  SAY "6. SRU INDENTURE TO:"
  @ 14, 21  GET  INDT_NSN6
  @ 14, 38  SAY "QPL:"
  @ 14, 43  GET  QPL6
  @ 16,  0  SAY "7. SRU INDENTURE TO:"
  @ 16, 21  GET  INDT_NSN7
  @ 16, 38  SAY "QPL:"
  @ 16, 43  GET  QPL7
  @ 18,  0  SAY "8. SRU INDENTURE TO:"
  @ 18, 21  GET  INDT_NSN8
  @ 18, 38  SAY "QPL:"
  @ 18, 43  GET  QPL8
  READ
ENDIF
LOOP
ELSE
  LOOP
ENDIF
ENDIF
CLEAR
@ 1,  0  SAY "NSN:"
@ 1,  4  GET  NSN

```

```

@ 1, 21 SAY "WUC:"
@ 1, 25 SAY WUC
@ 1, 31 SAY "NOUN:"
@ 1, 36 SAY NOUN
@ 1, 62 SAY "COST:"
@ 1, 67 SAY COST
@ 3, 0 SAY "COMP:"
@ 3, 5 SAY COMP
@ 3, 9 SAY "STOCK:"
@ 3, 15 SAY STK
@ 3, 19 SAY "QPA:"
@ 3, 24 SAY QPA
@ 4, 15 SAY "MIN QPA:"
@ 4, 24 SAY MQPA
@ 3, 27 SAY "RCT:"
@ 3, 32 SAY BRCT
@ 3, 39 SAY "NRTS:"
@ 3, 45 SAY NRTS_RT
@ 3, 51 SAY "DEPOT:"
@ 3, 57 SAY DEPOT
@ 5, 45 SAY "WADJ:"
@ 5, 51 SAY WADJ PICTURE "99.99"
@ 3, 60 SAY "DMD RATE:"
@ 3, 70 SAY DMD_RATE
@ 4, 60 SAY "DEMANDS:"
@ 4, 69 GET BASE1DMD
@ 5, 63 SAY "MTBD:"
@ 5, 69 GET MTBD
@ 6, 60 SAY "FH CODE:"
@ 6, 69 GET HOURS
@ 5, 0 SAY "PERCENT APPLICATION:"
IF LEN(TRIM(TAPP1))>0
  @ 6, 0 SAY TAPP1+[:]
  @ 6, 6 SAY APP1
ENDIF
IF LEN(TRIM(TAPP2))>0
  @ 6, 13 SAY TAPP2+[:]
  @ 6, 19 SAY APP2
ENDIF
IF LEN(TRIM(TAPP3))>0
  @ 6, 26 SAY TAPP3+[:]
  @ 6, 32 SAY APP3
ENDIF
IF LEN(TRIM(TAPP4))>0
  @ 7, 0 SAY TAPP4+[:]
  @ 7, 6 SAY APP4
ENDIF
IF LEN(TRIM(TAPP5))>0
  @ 7, 13 SAY TAPP5+[:]
  @ 7, 19 SAY APP5
ENDIF
IF LEN(TRIM(TAPP6))>0
  @ 7, 26 SAY TAPP6+[:]
  @ 7, 32 SAY APP6
ENDIF
IF LEN(TRIM(TAPP7))>0
  @ 8, 0 SAY TAPP7+[:]

```

```

      @ 8, 6 SAY APP7
    ENDIF
    IF LEN(TRIM(TAPP8))>0
      @ 8, 13 SAY TAPP8+[:]
      @ 8, 19 SAY APP8
    ENDIF
    IF LEN(TRIM(TAPP9))>0
      @ 8, 26 SAY TAPP9+[:]
      @ 8, 32 SAY APP9
    ENDIF
    IF LEN(TRIM(TAPP10))>0
      @ 9, 0 SAY TAPP10+[:]
      @ 9, 6 SAY APP10
    ENDIF
    IF LEN(TRIM(TAPP11))>0
      @ 9, 13 SAY TAPP11+[:]
      @ 9, 19 SAY APP11
    ENDIF
    IF LEN(TRIM(TAPP12))>0
      @ 9, 26 SAY TAPP12+[:]
      @ 9, 32 SAY APP12
    ENDIF
    @ 11, 10 SAY "LRU/SRU:"
    @ 11, 19 SAY LRU_SRU
    @ 11, 23 SAY "MAINT (0-RR,1-RRR):"
    @ 11, 43 SAY RR_RRR
    @ 16, 0 SAY "CURRENTY DELETED:"
    @ 16, 18 SAY DEL
    @ 16, 21 SAY "REVIEW:"
    @ 16, 29 GET REVIEWED
    @ 17, 0 SAY "REMARKS:"
    @ 18, 0 GET REMARK
    @ 19, 0 GET REMARKS
  READ
ENDDO
SET SAFE ON
SET TALK OFF
RETURN
ENDIF

DO WHILE .T.
  CLEAR
  @ 10, 0
  STORE " " TO NSN1
  @10,2 SAY;
  " NSN TO (D)EL / (R)ECOVER / (A)DD / EDIT ";
  GET NSN1 PICTURE "@!"
  @11,10 SAY [(P)REVIOUS / (N)EXT]
  READ
  CLEAR
  IF NSN1 = " "
    RETURN
  ENDIF

DO CASE

CASE NSN1 = "D "

```

```

CLEAR
@ 10, 0
STORE " " TO NSN1
@10,2 SAY;
    " NSN TO DELETE ";
    GET NSN1 PICTURE "@!"
READ
IF NSN1="P"
    SEEK TRIM(NSN2)
    @5,0
    @5,10 SAY "THE NSN IS "+NSN2
ELSE
    SEEK TRIM(NSN1)
ENDIF
IF .NOT. FOUND ()
    @15,2 SAY;
        " NSN NOT FOUND "
    LOOP
ENDIF

STORE " " TO MCONFIRM
CLEAR
@ 10,0
@10,10 SAY "ARE YOU SURE YOU WANT TO DELETE THIS NSN ? (Y/N)";
    GET MCONFIRM PICTURE "@!"
READ
IF .NOT. (MCONFIRM="N")
    DELETE
    REPLACE DEL WITH "Y"
    @ 23,15 SAY TRIM(NSN1) +;
        " HAS BEEN DELETED -- PRESS ANY KEY TO CONTINUE"
    WAIT " "
ENDIF
LOOP

CASE NSN1 = "R "

CLEAR
STORE " " TO NSN1
@10,2 SAY;
    " NSN TO RECOVER";
    GET NSN1 PICTURE "@!"
READ
SEEK TRIM(NSN1)
IF .NOT. FOUND ()
    @15,2 SAY;
        " NSN NOT FOUND "
    LOOP
ENDIF
CLEAR
STORE " " TO MCONFIRM
@ 10,0
@10,10 SAY "ARE YOU SURE YOU WANT TO RECOVER THIS NSN ? (Y/N)";
    GET MCONFIRM PICTURE "@!"
READ

```

```
IF .NOT. (MCONFIRM="N")
```

```
    RECALL
```

```
    REPLACE DEL WITH "N"
```

```
    @ 12,15 SAY TRIM (NSN) +;
```

```
        " has been recovered -- Press any key to continue"
```

```
    WAIT ""
```

```
ENDIF
```

```
LOOP
```

```
CASE NSN1 = "A "
```

```
DO WHILE .NOT. (NSN1=" ")
```

```
    CLEAR
```

```
    STORE " " TO NSN1
```

```
    CLEAR
```

```
    @ 10, 0
```

```
    @ 10,10 SAY "Enter the NSN for the new entry ";
```

```
        GET NSN1 PICTURE "@!";
```

```
    READ
```

```
    IF NSN1 = " "
```

```
        LOOP
```

```
    ENDIF
```

```
    SEEK NSN1
```

```
    NSN2=NSN1
```

```
    IF FOUND ()
```

```
        CLEAR
```

```
        @ 10,0
```

```
        @ 10,10 SAY TRIM (NSN1) +;
```

```
            " is already on file -- Press any key to continue"
```

```
        WAIT ""
```

```
    ELSE
```

```
        APPEND BLANK
```

```
        REPLACE NSN WITH NSN1,QPA WITH 1,MQPA WITH 1,CANN_IND WITH
```

```
0
```

```
        REPLACE LRU_SRU WITH "L",NRTS_RT WITH 1.00,BRCT WITH 2.00
```

```
        REPLACE RR_RRR WITH 0,APPL WITH 1.00,WADJ WITH 1.0
```

```
        REPLACE REVIEWED WITH "N"
```

```
        REPLACE DEL WITH "N"
```

```
        IND='1'
```

```
        DO WHILE VAL(IND)<21
```

```
            REPLACE APP&IND WITH 1
```

```
            IND=TRIM(LTRIM(STR(VAL(IND)+1)))
```

```
        ENDDO
```

```
    ENDIF
```

```
    CLEAR
```

```
    SET COLOR TO G+/N,W+/B
```

```
    @ 1, 0 SAY "NSN:"
```

```
    @ 1, 62 SAY "COST:"
```

```
    @ 3, 0 SAY "COMP:"
```

```
    @ 3, 9 SAY "STOCK:"
```

```
    @ 3, 19 SAY "QPA:"
```

```
    @ 4, 15 SAY "MIN QPA:"
```

```
    @ 3, 27 SAY "RCT:"
```

```
    @ 3, 39 SAY "NRTS:"
```

```
    @ 3, 60 SAY "DMD RATE:"
```



```

@ 5, 45 SAY "WADJ:"
@ 11, 10 SAY "LRU/SRU:"
@ 11, 23 SAY "MAINT (0-RR,1-RRR):"
SET COLOR TO R+/N
@ 1, 4 GET NSN
@ 1, 21 SAY "WUC:"
@ 1, 25 GET WUC
@ 1, 31 SAY "NOUN:"
@ 1, 36 GET NOUN
@ 1, 67 GET COST
@ 3, 5 GET COMP
@ 3, 15 GET STK
@ 3, 24 GET QPA
@ 4, 24 GET MQPA
@ 3, 32 GET BRCT
@ 3, 45 GET NRTS_RT
@ 3, 51 SAY "DEPOT:"
@ 3, 57 GET DEPOT
@ 5, 51 GET WADJ PICTURE "99.99"
@ 3, 70 GET DMD_RATE
@ 4, 60 SAY "DEMANDS:"
@ 4, 69 GET BASE1DMD
@ 5, 63 SAY "MTBD:"
@ 5, 69 GET MTBD
@ 6, 60 SAY "FH CODE:"
@ 6, 69 GET HOURS
SET COLOR TO G+/N,W+/B
@ 5, 0 SAY "PERCENT APPLICATION:"
IF LEN(TRIM(TAPP1))>0
    @ 6, 0 SAY TAPP1+[:]
    @ 6, 6 GET APP1
ENDIF
IF LEN(TRIM(TAPP2))>0
    @ 6, 13 SAY TAPP2+[:]
    @ 6, 19 GET APP2
ENDIF
IF LEN(TRIM(TAPP3))>0
    @ 6, 26 SAY TAPP3+[:]
    @ 6, 32 GET APP3
ENDIF
IF LEN(TRIM(TAPP4))>0
    @ 7, 0 SAY TAPP4+[:]
    @ 7, 6 GET APP4
ENDIF
IF LEN(TRIM(TAPP5))>0
    @ 7, 13 SAY TAPP5+[:]
    @ 7, 19 GET APP5
ENDIF
IF LEN(TRIM(TAPP6))>0
    @ 7, 26 SAY TAPP6+[:]
    @ 7, 32 GET APP6
ENDIF
IF LEN(TRIM(TAPP7))>0
    @ 8, 0 SAY TAPP7+[:]
    @ 8, 6 GET APP7
ENDIF
IF LEN(TRIM(TAPP8))>0

```

```

      @ 8, 13 SAY TAPP8[:]
      @ 8, 19 GET APP8
ENDIF
IF LEN(TRIM(TAPP9))>0
      @ 8, 26 SAY TAPP9[:]
      @ 8, 32 GET APP9
ENDIF
IF LEN(TRIM(TAPP10))>0
      @ 9, 0 SAY TAPP10[:]
      @ 9, 6 GET APP10
ENDIF
IF LEN(TRIM(TAPP11))>0
      @ 9, 13 SAY TAPP11[:]
      @ 9, 19 GET APP11
ENDIF
IF LEN(TRIM(TAPP12))>0
      @ 9, 26 SAY TAPP12[:]
      @ 9, 32 GET APP12
ENDIF
SET COLOR TO W+/N,W+/B
@ 11, 19 GET LRU_SRU
@ 11, 43 GET RR_RRR
@ 13, 10 SAY [ IF PART IS SRU, INDENTURES ARE SHOWN ON
FOLLOWING SCREEN]
@ 15, 0 SAY "CURRENTY DELETED:"
@ 15, 18 GET DEL
@ 15, 21 SAY "REVIEW:"
@ 15, 29 GET REVIEWED
@ 17, 0 SAY "REMARKS:"
@ 18, 0 GET REMARK
@ 19, 0 GET REMARKS
CLEAR TYPEAHEAD
READ
IF LRU_SRU="S"
  CLEAR
  @ 4, 0 SAY "1. SRU INDENTURE TO:"
  @ 4, 21 GET INDT_NSN1
  @ 4, 38 SAY "QPL:"
  @ 4, 43 GET QPL1
  @ 6, 0 SAY "2. SRU INDENTURE TO:"
  @ 6, 21 GET INDT_NSN2
  @ 6, 38 SAY "QPL:"
  @ 6, 43 GET QPL2
  @ 8, 0 SAY "3. SRU INDENTURE TO:"
  @ 8, 21 GET INDT_NSN3
  @ 8, 38 SAY "QPL:"
  @ 8, 43 GET QPL3
  @ 10, 0 SAY "4. SRU INDENTURE TO:"
  @ 10, 21 GET INDT_NSN4
  @ 10, 38 SAY "QPL:"
  @ 10, 43 GET QPL4
  @ 12, 0 SAY "5. SRU INDENTURE TO:"
  @ 12, 21 GET INDT_NSN5
  @ 12, 38 SAY "QPL:"
  @ 12, 43 GET QPL5
  @ 14, 0 SAY "6. SRU INDENTURE TO:"
  @ 14, 21 GET INDT_NSN6

```

```

      @ 14, 38 SAY "QPL:"
      @ 14, 43 GET  QPL6
      @ 16,  0 SAY "7. SRU INDENTURE TO:"
      @ 16, 21 GET  INDT_NSN7
      @ 16, 38 SAY "QPL:"
      @ 16, 43 GET  QPL7
      @ 18,  0 SAY "8. SRU INDENTURE TO:"
      @ 18, 21 GET  INDT_NSN8
      @ 18, 38 SAY "QPL:"
      @ 18, 43 GET  QPL8
      READ
    ENDIF

    IF DEL="Y"
      DELETE
    ELSE
      RECALL
    ENDIF
  ENDDO
LOOP

CASE NSN1="P "
  SEEK TRIM(NSN2)
  IF .NOT. FOUND()
    @ 10, 5 SAY "PREVIOUS STOCK NUMBER WAS NOT ADDED. PLEASE HIT
ANY KEY TO CONTINUE."
    WAIT " "
    LOOP
  ENDIF
  NSN1=NSN
  NSN2=NSN1
  CLEAR
  SET COLOR TO G+/N,W+/B
  @ 1,  0 SAY "NSN:"
  @ 1, 62 SAY "COST:"
  @ 3,  0 SAY "COMP:"
  @ 3,  9 SAY "STOCK:"
  @ 3, 19 SAY "QPA:"
  @ 4, 15 SAY "MIN QPA:"
  @ 3, 27 SAY "RCT:"
  @ 3, 39 SAY "NRTS:"
  @ 3, 60 SAY "DMD RATE:"
  @ 5, 45 SAY "WADJ:"
  @ 11, 10 SAY "LRU/SRU:"
  @ 11, 23 SAY "MAINT (0-RR,1-RRR):"
  SET COLOR TO R+/N
  @ 1,  4 GET  NSN
  @ 1, 21 SAY "WUC:"
  @ 1, 25 GET  WUC
  @ 1, 31 SAY "NOUN:"
  @ 1, 36 GET  NOUN
  @ 1, 67 GET  COST
  @ 3,  5 GET  COMP
  @ 3, 15 GET  STK
  @ 3, 24 GET  QPA
  @ 4, 24 GET  MQPA

```

```

@ 3, 32 GET BRCT
@ 3, 45 GET NRTS_RT
@ 3, 51 SAY "DEPOT:"
@ 3, 57 GET DEPOT
@ 5, 51 GET WADJ PICTURE "99.99"
@ 3, 70 GET DMD_RATE
@ 4, 60 SAY "DEMANDS:"
@ 4, 69 GET BASE1DMD
@ 5, 63 SAY "MTBD:"
@ 5, 69 GET MTBD
@ 6, 60 SAY "FH CODE:"
@ 6, 69 GET HOURS
SET COLOR TO G+/N,W+/B
@ 5, 0 SAY "PERCENT APPLICATION:"
IF LEN(TRIM(TAPP1))>0
    @ 6, 0 SAY TAPP1+[:]
    @ 6, 6 GET APP1
ENDIF
IF LEN(TRIM(TAPP2))>0
    @ 6, 13 SAY TAPP2+[:]
    @ 6, 19 GET APP2
ENDIF
IF LEN(TRIM(TAPP3))>0
    @ 6, 26 SAY TAPP3+[:]
    @ 6, 32 GET APP3
ENDIF
IF LEN(TRIM(TAPP4))>0
    @ 7, 0 SAY TAPP4+[:]
    @ 7, 6 GET APP4
ENDIF
IF LEN(TRIM(TAPP5))>0
    @ 7, 13 SAY TAPP5+[:]
    @ 7, 19 GET APP5
ENDIF
IF LEN(TRIM(TAPP6))>0
    @ 7, 26 SAY TAPP6+[:]
    @ 7, 32 GET APP6
ENDIF
IF LEN(TRIM(TAPP7))>0
    @ 8, 0 SAY TAPP7+[:]
    @ 8, 6 GET APP7
ENDIF
IF LEN(TRIM(TAPP8))>0
    @ 8, 13 SAY TAPP8+[:]
    @ 8, 19 GET APP8
ENDIF
IF LEN(TRIM(TAPP9))>0
    @ 8, 26 SAY TAPP9+[:]
    @ 8, 32 GET APP9
ENDIF
IF LEN(TRIM(TAPP10))>0
    @ 9, 0 SAY TAPP10+[:]
    @ 9, 6 GET APP10
ENDIF
IF LEN(TRIM(TAPP11))>0
    @ 9, 13 SAY TAPP11+[:]
    @ 9, 19 GET APP11

```

```

ENDIF
IF LEN(TRIM(TAPP12))>0
  @ 9, 26 SAY TAPP12+[:]
  @ 9, 32 GET APP12
ENDIF
SET COLOR TO W+/N,W+/B
@ 11, 19 GET LRU_SRU
@ 11, 43 GET RR_RRR
@ 13, 10 SAY [ IF PART IS SRU, INDENTURES ARE SHOWN ON
FOLLOWING SCREEN]
@ 15, 0 SAY "CURRENTY DELETED:"
@ 15, 18 GET DEL
@ 15, 21 SAY "REVIEW:"
@ 15, 29 GET REVIEWED
@ 17, 0 SAY "REMARKS:"
@ 18, 0 GET REMARK
@ 19, 0 GET REMARKS
CLEAR TYPEAHEAD
READ
IF LRU_SRU="S"
  CLEAR
  @ 4, 0 SAY "1. SRU INDENTURE TO:"
  @ 4, 21 GET INDT_NSN1
  @ 4, 38 SAY "QPL:"
  @ 4, 43 GET QPL1
  @ 6, 0 SAY "2. SRU INDENTURE TO:"
  @ 6, 21 GET INDT_NSN2
  @ 6, 38 SAY "QPL:"
  @ 6, 43 GET QPL2
  @ 8, 0 SAY "3. SRU INDENTURE TO:"
  @ 8, 21 GET INDT_NSN3
  @ 8, 38 SAY "QPL:"
  @ 8, 43 GET QPL3
  @ 10, 0 SAY "4. SRU INDENTURE TO:"
  @ 10, 21 GET INDT_NSN4
  @ 10, 38 SAY "QPL:"
  @ 10, 43 GET QPL4
  @ 12, 0 SAY "5. SRU INDENTURE TO:"
  @ 12, 21 GET INDT_NSN5
  @ 12, 38 SAY "QPL:"
  @ 12, 43 GET QPL5
  @ 14, 0 SAY "6. SRU INDENTURE TO:"
  @ 14, 21 GET INDT_NSN6
  @ 14, 38 SAY "QPL:"
  @ 14, 43 GET QPL6
  @ 16, 0 SAY "7. SRU INDENTURE TO:"
  @ 16, 21 GET INDT_NSN7
  @ 16, 38 SAY "QPL:"
  @ 16, 43 GET QPL7
  @ 18, 0 SAY "8. SRU INDENTURE TO:"
  @ 18, 21 GET INDT_NSN8
  @ 18, 38 SAY "QPL:"
  @ 18, 43 GET QPL8
  READ
ENDIF
IF DEL="Y"
  DELETE

```

```

ELSE
    RECALL
ENDIF
LOOP

```

```

CASE NSN1="N "
    IF EOF()
        CLEAR
        @10,5 SAY [THIS IS THE END OF THE FILE]
        @12,5 SAY [PLEASE STRIKE ANY KEY TO CONTINUE]
        WAIT " "
        SKIP -1
        NSN1=NSN
        NSN2=NSN1
        LOOP
    ENDIF
    SKIP
    IF EOF()
        CLEAR
        @10,5 SAY [THIS IS THE END OF THE FILE]
        @12,5 SAY [PLEASE STRIKE ANY KEY TO CONTINUE]
        WAIT " "
        SKIP -1
        NSN1=NSN
        NSN2=NSN1
        LOOP
    ENDIF
    NSN1=NSN
    NSN2=NSN1
    CLEAR
    SET COLOR TO G+/N,W+/B
    @ 1, 0 SAY "NSN:"
    @ 1, 62 SAY "COST:"
    @ 3, 0 SAY "COMP:"
    @ 3, 9 SAY "STOCK:"
    @ 3, 19 SAY "QPA:"
    @ 4, 15 SAY "MIN QPA:"
    @ 3, 27 SAY "RCT:"
    @ 3, 39 SAY "NRTS:"
    @ 3, 60 SAY "DMD RATE:"
    @ 5, 45 SAY "WADJ:"
    @ 11, 10 SAY "LRU/SRU:"
    @ 11, 23 SAY "MAINT (0-RR,1-RRR):"
    SET COLOR TO R+/N
    @ 1, 4 GET NSN
    @ 1, 21 SAY "WUC:"
    @ 1, 25 GET WUC
    @ 1, 31 SAY "NOUN:"
    @ 1, 36 GET NOUN
    @ 1, 67 GET COST
    @ 3, 5 GET COMP
    @ 3, 15 GET STK
    @ 3, 24 GET QPA
    @ 4, 24 GET MQPA
    @ 3, 32 GET BRCT
    @ 3, 45 GET NRTS_RT

```

```

@ 3, 51 SAY "DEPOT:"
@ 3, 57 GET DEPOT
@ 5, 51 GET WADJ PICTURE "99.99"
@ 3, 70 GET DMD_RATE
@ 4, 60 SAY "DEMANDS:"
@ 4, 69 GET BASE1DMD
@ 5, 63 SAY "MTBD:"
@ 5, 69 GET MTBD
@ 6, 60 SAY "FH CODE:"
@ 6, 69 GET HOURS
SET COLOR TO G+/N,W+/B
@ 5, 0 SAY "PERCENT APPLICATION:"
IF LEN(TRIM(TAPP1))>0
    @ 6, 0 SAY TAPP1+[:]
    @ 6, 6 GET APP1
ENDIF
IF LEN(TRIM(TAPP2))>0
    @ 6, 13 SAY TAPP2+[:]
    @ 6, 19 GET APP2
ENDIF
IF LEN(TRIM(TAPP3))>0
    @ 6, 26 SAY TAPP3+[:]
    @ 6, 32 GET APP3
ENDIF
IF LEN(TRIM(TAPP4))>0
    @ 7, 0 SAY TAPP4+[:]
    @ 7, 6 GET APP4
ENDIF
IF LEN(TRIM(TAPP5))>0
    @ 7, 13 SAY TAPP5+[:]
    @ 7, 19 GET APP5
ENDIF
IF LEN(TRIM(TAPP6))>0
    @ 7, 26 SAY TAPP6+[:]
    @ 7, 32 GET APP6
ENDIF
IF LEN(TRIM(TAPP7))>0
    @ 8, 0 SAY TAPP7+[:]
    @ 8, 6 GET APP7
ENDIF
IF LEN(TRIM(TAPP8))>0
    @ 8, 13 SAY TAPP8+[:]
    @ 8, 19 GET APP8
ENDIF
IF LEN(TRIM(TAPP9))>0
    @ 8, 26 SAY TAPP9+[:]
    @ 8, 32 GET APP9
ENDIF
IF LEN(TRIM(TAPP10))>0
    @ 9, 0 SAY TAPP10+[:]
    @ 9, 6 GET APP10
ENDIF
IF LEN(TRIM(TAPP11))>0
    @ 9, 13 SAY TAPP11+[:]
    @ 9, 19 GET APP11
ENDIF
IF LEN(TRIM(TAPP12))>0

```

```

      @ 9, 26 SAY TAPP12+[:]
      @ 9, 32 GET APP12
ENDIF
SET COLOR TO W+/N,W+/B
@ 11, 19 GET LRU_SRU
@ 11, 43 GET RR_RRR
@ 13, 10 SAY [ IF PART IS SRU, INDENTURES ARE SHOWN ON
FOLLOWING SCREEN]
@ 15, 0 SAY "CURRENTY DELETED:"
@ 15, 18 GET DEL
@ 15, 21 SAY "REVIEW:"
@ 15, 29 GET REVIEWED
@ 17, 0 SAY "REMARKS:"
@ 18, 0 GET REMARK
@ 19, 0 GET REMARKS
CLEAR TYPEAHEAD
READ
IF LRU_SRU="S"
  CLEAR
  @ 4, 0 SAY "1. SRU INDENTURE TO:"
  @ 4, 21 GET INDT_NSN1
  @ 4, 38 SAY "QPL:"
  @ 4, 43 GET QPL1
  @ 6, 0 SAY "2. SRU INDENTURE TO:"
  @ 6, 21 GET INDT_NSN2
  @ 6, 38 SAY "QPL:"
  @ 6, 43 GET QPL2
  @ 8, 0 SAY "3. SRU INDENTURE TO:"
  @ 8, 21 GET INDT_NSN3
  @ 8, 38 SAY "QPL:"
  @ 8, 43 GET QPL3
  @ 10, 0 SAY "4. SRU INDENTURE TO:"
  @ 10, 21 GET INDT_NSN4
  @ 10, 38 SAY "QPL:"
  @ 10, 43 GET QPL4
  @ 12, 0 SAY "5. SRU INDENTURE TO:"
  @ 12, 21 GET INDT_NSN5
  @ 12, 38 SAY "QPL:"
  @ 12, 43 GET QPL5
  @ 14, 0 SAY "6. SRU INDENTURE TO:"
  @ 14, 21 GET INDT_NSN6
  @ 14, 38 SAY "QPL:"
  @ 14, 43 GET QPL6
  @ 16, 0 SAY "7. SRU INDENTURE TO:"
  @ 16, 21 GET INDT_NSN7
  @ 16, 38 SAY "QPL:"
  @ 16, 43 GET QPL7
  @ 18, 0 SAY "8. SRU INDENTURE TO:"
  @ 18, 21 GET INDT_NSN8
  @ 18, 38 SAY "QPL:"
  @ 18, 43 GET QPL8
  READ
ENDIF
IF DEL="Y"
  DELETE
ELSE
  RECALL

```



```
ENDIF
LOOP
```

```
OTHERWISE
  SEEK TRIM(NSN1)
  NSN2=NSN1
  IF .NOT. FOUND ()
    CLEAR
    STORE " " TO ANSW
    @ 10,15 SAY TRIM(NSN1) +;
      " NOT FOUND"
    @ 15,15 SAY "DO YOU WANT TO ADD THIS NSN ? (Y/N)";
      GET ANSW PICTURE "@!"
    READ
    IF ANSW = "N"
      LOOP
    ENDIF
    APPEND BLANK
    REPLACE NSN WITH NSN1,QPA WITH 1,MQPA WITH 1,CANN_IND WITH 0
    REPLACE LRU_SRU WITH "L",NRTS_RT WITH 1.00,BRCT WITH 2.00
    REPLACE RR_RRR WITH 0,APPL WITH 1.00,WADJ WITH 1.0
    REPLACE REVIEWED WITH "N"
    REPLACE DEL WITH "N"
    IND='1'
    DO WHILE VAL(IND)<21
      REPLACE APP&IND WITH 1
      IND=TRIM(LTRIM(STR(VAL(IND)+1)))
    ENDDO
  ENDIF
  CLEAR
  SET COLOR TO G+/N,W+/B
  @ 1, 0 SAY "NSN:"
  @ 1, 62 SAY "COST:"
  @ 3, 0 SAY "COMP:"
  @ 3, 9 SAY "STOCK:"
  @ 3, 19 SAY "QPA:"
  @ 4, 15 SAY "MIN QPA:"
  @ 3, 27 SAY "RCT:"
  @ 3, 39 SAY "NRTS:"
  @ 3, 60 SAY "DMD RATE:"
  @ 5, 45 SAY "WADJ:"
  @ 11, 10 SAY "LRU/SRU:"
  @ 11, 23 SAY "MAINT (0-RR,1-RRR):"
  SET COLOR TO R+/N
  @ 1, 4 GET NSN
  @ 1, 21 SAY "WUC:"
  @ 1, 25 GET WUC
  @ 1, 31 SAY "NOUN:"
  @ 1, 36 GET NOUN
  @ 1, 67 GET COST
  @ 3, 5 GET COMP
  @ 3, 15 GET STK
  @ 3, 24 GET QPA
  @ 4, 24 GET MQPA
  @ 3, 32 GET BRCT
  @ 3, 45 GET NRTS_RT
```

```

@ 3, 51 SAY "DEPOT:"
@ 3, 57 GET DEPOT
@ 5, 51 GET WADJ PICTURE "99.99"
@ 3, 70 GET DMD_RATE
@ 4, 60 SAY "DEMANDS:"
@ 4, 69 GET BASE1DMD
@ 5, 63 SAY "MTBD:"
@ 5, 69 GET MTBD
@ 6, 60 SAY "FH CODE:"
@ 6, 69 GET HOURS
SET COLOR TO G+/N,W+/B
@ 5, 0 SAY "PERCENT APPLICATION:"
IF LEN(TRIM(TAPP1))>0
    @ 6, 0 SAY TAPP1+[:]
    @ 6, 6 GET APP1
ENDIF
IF LEN(TRIM(TAPP2))>0
    @ 6, 13 SAY TAPP2+[:]
    @ 6, 19 GET APP2
ENDIF
IF LEN(TRIM(TAPP3))>0
    @ 6, 26 SAY TAPP3+[:]
    @ 6, 32 GET APP3
ENDIF
IF LEN(TRIM(TAPP4))>0
    @ 7, 0 SAY TAPP4+[:]
    @ 7, 6 GET APP4
ENDIF
IF LEN(TRIM(TAPP5))>0
    @ 7, 13 SAY TAPP5+[:]
    @ 7, 19 GET APP5
ENDIF
IF LEN(TRIM(TAPP6))>0
    @ 7, 26 SAY TAPP6+[:]
    @ 7, 32 GET APP6
ENDIF
IF LEN(TRIM(TAPP7))>0
    @ 8, 0 SAY TAPP7+[:]
    @ 8, 6 GET APP7
ENDIF
IF LEN(TRIM(TAPP8))>0
    @ 8, 13 SAY TAPP8+[:]
    @ 8, 19 GET APP8
ENDIF
IF LEN(TRIM(TAPP9))>0
    @ 8, 26 SAY TAPP9+[:]
    @ 8, 32 GET APP9
ENDIF
IF LEN(TRIM(TAPP10))>0
    @ 9, 0 SAY TAPP10+[:]
    @ 9, 6 GET APP10
ENDIF
IF LEN(TRIM(TAPP11))>0
    @ 9, 13 SAY TAPP11+[:]
    @ 9, 19 GET APP11
ENDIF
IF LEN(TRIM(TAPP12))>0

```

```

      @ 9, 26 SAY TAPP12+[:]
      @ 9, 32 GET APP12
ENDIF
SET COLOR TO W+/N,W+/B
@ 11, 19 GET LRU_SRU
@ 11, 43 GET RR_RRR
@ 13, 10 SAY [ IF PART IS SRU, INDENTURES ARE SHOWN ON
FOLLOWING SCREEN]
@ 15, 0 SAY "CURRENTY DELETED:"
@ 15, 18 GET DEL
@ 15, 21 SAY "REVIEW:"
@ 15, 29 GET REVIEWED
@ 17, 0 SAY "REMARKS:"
@ 18, 0 GET REMARK
@ 19, 0 GET REMARKS
CLEAR TYPEAHEAD
READ
IF LRU_SRU="S"
  CLEAR
  @ 4, 0 SAY "1. SRU INDENTURE TO:"
  @ 4, 21 GET INDT_NSN1
  @ 4, 38 SAY "QPL:"
  @ 4, 43 GET QPL1
  @ 6, 0 SAY "2. SRU INDENTURE TO:"
  @ 6, 21 GET INDT_NSN2
  @ 6, 38 SAY "QPL:"
  @ 6, 43 GET QPL2
  @ 8, 0 SAY "3. SRU INDENTURE TO:"
  @ 8, 21 GET INDT_NSN3
  @ 8, 38 SAY "QPL:"
  @ 8, 43 GET QPL3
  @ 10, 0 SAY "4. SRU INDENTURE TO:"
  @ 10, 21 GET INDT_NSN4
  @ 10, 38 SAY "QPL:"
  @ 10, 43 GET QPL4
  @ 12, 0 SAY "5. SRU INDENTURE TO:"
  @ 12, 21 GET INDT_NSN5
  @ 12, 38 SAY "QPL:"
  @ 12, 43 GET QPL5
  @ 14, 0 SAY "6. SRU INDENTURE TO:"
  @ 14, 21 GET INDT_NSN6
  @ 14, 38 SAY "QPL:"
  @ 14, 43 GET QPL6
  @ 16, 0 SAY "7. SRU INDENTURE TO:"
  @ 16, 21 GET INDT_NSN7
  @ 16, 38 SAY "QPL:"
  @ 16, 43 GET QPL7
  @ 18, 0 SAY "8. SRU INDENTURE TO:"
  @ 18, 21 GET INDT_NSN8
  @ 18, 38 SAY "QPL:"
  @ 18, 43 GET QPL8
  READ
ENDIF
IF DEL="Y"
  DELETE
ELSE

```

```

        RECALL
      ENDIF

```

```

    ENDCASE
  ENDDO

```

```

RETURN

```

```

*: EOF: BREVIEW.PRG

```

```

*: *****
*
*:
*:      Program: BUTIL.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/25/95    14:38
*:
*:      Called by: BASEAID.PRG
*:
*:      Calls: MYREAD.PRG
*:             : READDMMAS.PRG
*:
*:      Uses: A:&TXT.DBF
*:            : BLNK.DBF
*:
*:      Documented: 07/16/95 at 04:44          SNAP! version
3.10a
*: *****
*

```

```

DO WHILE .T.

```

```

    * ---Display menu options, centered on the screen.
    *   draw menu border and print heading
    CLEAR
    @ 2, 0 TO 16,79 DOUBLE
    @ 3,25 SAY [U T I L I T I E S]
    @ 4,1 TO 4,78 DOUBLE
    * ---display detail lines
    @ 7,19 SAY [1. CONVERT DMAS FILE TO DATABASE FILE]
    @ 8,19 SAY [2. RESET ALL APPLICATION FRACTIONS TO 1.00]
    @ 9,19 SAY [3. RESET ALL REVIEWED FLAGS]
    @ 10,19 SAY [4. PACK DATABASE (ELIMINATE DELETED ITEMS)]
    @ 11,19 SAY [5. CHANGE PRINTER AVAILABILITY]
    @ 12,19 SAY [6. READ DMAS COMPUTED QTYs INTO STK COLUMN]
    @ 14, 19 SAY '0. EXIT'
    STORE 0 TO SELECTNUM
    @ 16,33 SAY " select      "
    @ 16,42 GET SELECTNUM PICTURE "9" RANGE 0,5
    READ

```

```

DO CASE

```

```

CASE SELECTNUM = 0
  RETURN

CASE SELECTNUM = 1
  * CONVERT DMAS TO BASEAID FILE

  STXT = [      ]
  TXT = [      ]
  CLEAR
  @ 10, 5 SAY "Which DMAS INPUT TEXT FILE should I use? " GET
STXT
  READ
  IF STXT = " "
    LOOP
  ENDIF
  CLEAR
  ANS=[ ]
  @ 10 ,5 SAY "IS IT ON A OR C DRIVE? (A/C)" GET ANS
  READ
  STXT = UPPER(LTRIM(TRIM(STXT)))
  TXT = [&STXT] + [.TXT]
  IF ANS="A"
    IF .NOT. FILE('A:&TXT')
      @ 12, 5 SAY "The file &TXT does not exist."
      SET CONFIRM OFF
      STORE ' ' TO WAIT_SUBST
      @ 23,0 SAY 'Press any key to continue...' GET WAIT_SUBST
      READ
      SET CONFIRM ON
      LOOP
    ENDIF
    APPE FROM A:&TXT SDF
  ELSE
    IF .NOT. FILE(TXT)
      @ 12, 5 SAY "The file &TXT does not exist."
      SET CONFIRM OFF
      STORE ' ' TO WAIT_SUBST
      @ 23,0 SAY 'Press any key to continue...' GET WAIT_SUBST
      READ
      SET CONFIRM ON
      LOOP
    ENDIF
  ENDIF
  DO MYREAD
  REPLACE ALL REVIEWED WITH "Y"
  REPLACE ALL HOURS WITH "A"

  GO TOP
  DO WHILE .NOT. EOF()
    IF LRU_SRU='S'
      REPLACE APP1 WITH 1.0
      IF WADJ>0.9
        SKIP
      LOOP
    ENDIF
    SNSN=NSN
    LRU=INDT_NSN1

```

```

        SEEK LRU
        IF FOUND()
            WA=WADJ
        ELSE
            WA=1
        ENDIF
        SEEK SNSN
        REPLACE WADJ WITH WA
    ENDIF
    SKIP
    LOOP
ENDDO

```

```

SELE 5
USE BLNK
SET SAFE OFF
ZAP
SET SAFE ON
SELE 1

```

```

SET CONFIRM ON

```

```

CASE SELECTNUM = 2
    * RESET ALL APPLICATION FRACTIONS TO 1.00
    GO TOP
    DO WHILE .NOT. EOF()
        IND='1'
        CLEAR
        NO=RECNO()
        NO=STR(NO)
        @ 20, 5 SAY "WORKING... " + NO
        DO WHILE VAL(IND) < 21
            REPLACE APP&IND WITH 1
            IND=TRIM(LTRIM(STR(VAL(IND)+1)))
        ENDDO
        SKIP
    ENDDO
    CLEAR

```

```

CASE SELECTNUM = 3
    * RESET ALL REVIEWED FLAGS
    CLEAR
    ANSR=[ ]
    @ 10, 5 SAY "WHAT DO YOU WANT REVIEWED FLAG RESET TO ? [Y/N] "
GET ANSR
    READ
    ANSR=UPPER(ANSR)
    SET TALK ON
    REPLACE ALL REVIEWED WITH ANSR
    SET TALK OFF

```

```

CASE SELECTNUM = 4
    * PACK DATABASE (ELIMINATE DELETED ITEMS)
    SET TALK ON
    SET DELETE OFF
    SET SAFE OFF

```

```

        PACK
        SET SAFE ON
        SET TALK OFF

CASE SELECTNUM = 5
    * CHANGE PRINTER STATUS
    CLEAR
    @ 10, 8 SAY [ARE YOU USING A PRINTER ?];
    GET PRNT PICTURE "@!"
    READ
    PRNT=UPPER(PRNT)

CASE SELECTNUM = 6
    DO READDMS
    SET PRINT OFF
    SET TALK OFF

ENDCASE

ENDDO T
RETURN
* EOF: UTIL.PRG

*: EOF: BUTIL.PRG

*: *****
*
*:
*:      Program: BWRSKNAME.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/24/95      17:21
*:
*:      Called by: BASEAID.PRG
*:
*:      Documented: 07/16/95 at 04:44                      SNAP! version
3.10a
*: *****
*
SELE N&PDB
SET TALK OFF
GO TOP
CLEAR
@ 3, 6 SAY "PLEASE GIVE THE FOLLOWING KIT IDENTIFIERS ?"
@ 5, 30 SAY "BASE NAME:"
@ 5, 42 GET TBASE1
@ 6, 32 SAY "KIT 1:"
@ 6, 38 GET TAPP1
@ 7, 32 SAY "KIT 2:"
@ 7, 38 GET TAPP2
@ 8, 32 SAY "KIT 3:"

```

```

@ 8, 38 GET TAPP3
@ 9, 32 SAY "KIT 4:"
@ 9, 38 GET TAPP4
@ 10, 32 SAY "KIT 5:"
@ 10, 38 GET TAPP5
@ 11, 32 SAY "KIT 6:"
@ 11, 38 GET TAPP6
@ 12, 32 SAY "KIT 7:"
@ 12, 38 GET TAPP7
@ 13, 32 SAY "KIT 8:"
@ 13, 38 GET TAPP8
@ 14, 32 SAY "KIT 9:"
@ 14, 38 GET TAPP9
@ 15, 32 SAY "KIT10:"
@ 15, 38 GET TAPP10
@ 16, 32 SAY "KIT11:"
@ 16, 38 GET TAPP11
@ 17, 32 SAY "KIT12:"
@ 17, 38 GET TAPP12
CLEAR TYPEAHEAD
READ
REPLACE BASE1 WITH TBASE1
REPLACE APP1 WITH TAPP1
REPLACE APP2 WITH TAPP2
REPLACE APP3 WITH TAPP3
REPLACE APP4 WITH TAPP4
REPLACE APP5 WITH TAPP5
REPLACE APP6 WITH TAPP6
REPLACE APP7 WITH TAPP7
REPLACE APP8 WITH TAPP8
REPLACE APP9 WITH TAPP9
REPLACE APP10 WITH TAPP10
REPLACE APP11 WITH TAPP11
REPLACE APP12 WITH TAPP12
GO TOP
SELE 1
RETURN
*: EOF: BWRSKNAME.PRG

```

```

* .*****
*
* :
* :      Program: DMV4OUT.PRG
* :
* :      System: disc-data-handler
* :      Author: t.lewis
* :      Copyright (c) 1994, t.lewis
* :      Last modified: 06/24/95      17:21
* :
* :      Called by: BBUILD.PRG
* :
* :      Uses: INDT.DBF
* :
* :      Indexes: LRU.NDX
* :

```



```

*:      Documented: 07/16/95 at 04:44                      SNAP! version
3.10a
*:*****
*
GO TOP

SET ALTE ON

?" 1 0.0 0.          VERSION 4.4  M1"
?" 1 4 7 8 10 15 20 25 30"
?"OPT"
?"      8 20"
?"      11 5 .80 "
?"      14"
?"BASE"
LINE1=BNAME+"      0.  0.  0.  0 1.0099.0030.000  1.0 99.0199.0
3.0100.01 0.  00.00 10"
?LINE1
?"ACFT"
LINE2=BNAME+"  24  99"
?LINE2
?"SRTS"
LINE3=BNAME+"  0.  1 0.0  8 0.0  99"
?LINE3
?"FLHR"
LINE4=BNAME+" 0.0  99"
?LINE4
?"TURN"
LINE5=BNAME+" 3.5  99"
?LINE5

*
* LRU
*
?"LRU"
DO WHILE .NOT.EOF()
  IF .NOT.(LRU_SRU="L")
    SKIP
    LOOP
  ENDIF
  LINE1=NSN+"      "+"1  "+STR(QPA,3)+STR(QPA,3)+"
"+STR(DMD_RATE,7,5)+STR(DMD_RATE,7,5)+STR(BRCT,5,2)+"
"+STR(NRTS_RT,4,2)
  LINE1=LINE1+" 000000000 0000 0000"
  LINE2=NSN+" 0000 0000 000000000 0000 0000000000000000 "+STR(COST,8)+"
"+WUC
  LINE2=LINE2+" "+STR(CANN_IND,1)
  ? LINE1
  ? LINE2
  SKIP
ENDDO

*
* APP FRAC
*
GO TOP
?"APPL"

```

```

DO WHILE .NOT.EOF()
  IF .NOT.(LRU_SRU="L")
    SKIP
    LOOP
  ENDIF
  LINE1=NSN+" "+BASE+STR(APP&APPNO,4,2)
  ?LINE1
  SKIP
ENDDO

```

```

*
*   VTM
*
GO TOP
?"VTM"

```

```

DO WHILE .NOT.EOF()
  IF .NOT.(LRU_SRU="L")
    SKIP
    LOOP
  ENDIF
  LINE1=NSN+" "+STR(RR_RRR,1,0)+" "+STR(WADJ,4,2)+" "+STR(WADJ,4,2)+"
  "+"1.00"+"      "+"      1"
  ?LINE1
  SKIP
ENDDO

```

```

*
*   SRU
*

```

```

GO TOP
?"SRU"
SELECT 2
USE INDT
SET SAFETY OFF
ZAP
SET SAFETY ON
SELECT TEMP

```

```

DO WHILE .NOT.EOF()
  IF .NOT.(LRU_SRU="S")
    SKIP
    LOOP
  ENDIF
  TSN=NSN
  LSN1=SUBSTR(INDT_NSN1,1,13)
  LSN2=SUBSTR(INDT_NSN2,1,13)
  LSN3=SUBSTR(INDT_NSN3,1,13)
  LSN4=SUBSTR(INDT_NSN4,1,13)
  LSN5=SUBSTR(INDT_NSN5,1,13)
  LSN6=SUBSTR(INDT_NSN6,1,13)
  LSN7=SUBSTR(INDT_NSN7,1,13)
  LSN8=SUBSTR(INDT_NSN8,1,13)
  IND='2'
  TQPL=QPA

```

```

SELECT INDT
APPEND BLANK
REPLACE SRU WITH TSN
REPLACE LRU WITH LSN1
REPLACE QPL WITH TEMP->QPL1
DO WHILE VAL(IND) < 9
    IF LEN(TRIM(LSN&IND)) > 0
        APPEND BLANK
        REPLACE SRU WITH TSN
        REPLACE LRU WITH LSN&IND
        REPLACE QPL WITH TEMP->QPL&IND
    ENDIF
    IND=TRIM(LTRIM(STR(VAL(IND)+1)))
ENDDO
SELE TEMP
LINE1=NSN+"      1  "+STR(QPA,3,0)+"
"+STR(DMD_RATE,7,5)+STR(DMD_RATE,7,5)+STR(BRCT,5,2)+"
"+STR(NRTS_RT,4,2)+" "+"0000"+"0000000000000000"
LINE2=NSN+" 0000 0000 0000000000 0000 0000000000000000 "+STR(COST,8)+"
"+WUC
LINE2=LINE2+" "+STR(CANN_IND,1)
?LINE1
?LINE2
SKIP
ENDDO

*
*INDT
*

SELE INDT
GO TOP
SET SAFE OFF
INDEX ON LRU TO LRU
SET SAFE ON
?"INDT"
IF RECCOUNT () >0
    LINE=LRU+"1L"
    ?LINE
    PLRU=LRU
    DO WHILE .NOT. EOF()
        IF .NOT. (LRU=PLRU)
            LINE=LRU+"1L"
            ?LINE
            PLRU=LRU
        ENDIF
        LINE1=SRU+" S"+STR(QPL,3)
        ?LINE1
        PSRU=TRIM(SRU)
        SELE TEMP
        SEEK TRIM(PLRU)
        IF .NOT. FOUND()
            MESSAGE=5
            CLEAR
            @ 5,33 SAY "*** WARNING ***"
            @ 7,0 SAY "SRU "+PSRU+"'s LRU "+PLRU+"HAS BEEN DELETED OR NOT
REVIEWED"

```

```

@ 9,0 SAY "THIS WILL CAUSE FATAL ERRORS IN PROCESSING"
STORE " " TO ANSW
@ 15,10 SAY "DO YOU WANT TO CONTINUE ? (Y/N)";
GET ANSW PICTURE "@!";
READ
IF ANSW = "N"
    USE
    SELE 1
    SET DELE OFF
    SET ALTE TO
    RETURN
ENDIF
ELSE
    IF RR_RRR<>1
        SELE 2
        REPLACE ERROR2 WITH "Y"
        MESSAGE=5
        SELE 3
    ENDIF
    IF LRU_SRU="S"
        SELE 2
        REPLACE ERROR2 WITH "L"
        MESSAGE=5
        SELE 3
    ENDIF
    TMDR=DMD_RATE
    SEEK PSRU
    SMDR=DMD_RATE
    IF SMDR>TMDR
        SET ALTE OFF
        CLEAR
        @10,1 SAY "WARNING: THE DEMAND RATE FOR SRU "+PSRU+" IS
GREATER THAN ITS PARENT"
        @11,1 SAY "LRUs "+PLRU+" DEMAND RATE. THIS SHOULD NOT
HAPPEN. CHECK THESE NSNs DEMAND RATES."
        STORE " " TO ANSW
        @ 15,10 SAY "DO YOU WANT TO CONTINUE ? (Y/N)";
        GET ANSW PICTURE "@!";
        READ
        IF ANSW = "N"
            USE
            SELE 1
            SET DELE OFF
            SET ALTE TO
            RETURN
        ENDIF
        SET ALTE ON
    ENDIF
ENDIF
SELE INDT
SKIP
ENDDO
ENDIF

*
*   STK LEVEL
*

```

```

SELE TEMP
GO TOP
?"STK"
DO WHILE .NOT.EOF()
    LINE1=NSN+" "+BASE+" "+STR(STK,3)
    ?LINE1
    SKIP
ENDDO

```

```

*
*   INDICATIVE DATA
*
GO TOP
?"INDC"
DO WHILE .NOT.EOF()
    NAME=LTRIM(NOUN)
    IF .NOT. (LEN(NAME)>0)
        WAR=WADJ*100
        WAR1=STR(WAR,4)
        NSN1=SUBSTR(NSN,1,13)
        LINE1=NSN1+" "+WUC+" "+COMP+WAR1
        ?LINE1
        SKIP
        LOOP
    ENDIF
    NAME=SUBSTR(NAME,1,10)
    WAR=WADJ*100
    WAR1=STR(WAR,4)
    NSN1=SUBSTR(NSN,1,13)
    LINE1=NSN1+" "+WUC+" "+NAME+COMP+WAR1
    ?LINE1
    SKIP
ENDDO
?"END"

```

```

SET ALTE OFF
*: EOF: DMV4OUT.PRG

```

```

*: *****
*
*:
*:   Program: DMV6OUT.PRG
*:
*:   System: disc-data-handler
*:   Author: t.lewis
*:   Copyright (c) 1994, t.lewis
*:   Last modified: 06/24/95    17:21
*:
*:   Called by: BBUILD.PRG
*:
*:   Uses: INDT.DBF
*:
*:   Indexes: LRU.NDX
*:
*:   Documented: 07/16/95 at 04:44
3.10a

```

SNAP! version


```

        LOOP
    ENDIF
    LINE1=NSN+" "+BASE+STR(APP&APPNO,4,2)
    ?LINE1
    SKIP
ENDDO

*
*   VTM
*
GO TOP
?"MISC"

DO WHILE .NOT.EOF()
    IF .NOT.(LRU_SRU="L")
        SKIP
        LOOP
    ENDIF
    LINE1=NSN+" "+STR(WADJ,4,2)+" "+STR(WADJ,4,2)+" "+"1.00"
    ?LINE1
    SKIP
ENDDO

*
*   SRU
*

GO TOP
?"SRU"
SELECT 2
USE INDT
SET SAFETY OFF
ZAP
SET SAFETY ON
SELECT TEMP

DO WHILE .NOT. EOF()
    IF .NOT.(LRU_SRU="S")
        SKIP
        LOOP
    ENDIF
    TSN=NSN
    LSN1=SUBSTR(INDT_NSN1,1,13)
    LSN2=SUBSTR(INDT_NSN2,1,13)
    LSN3=SUBSTR(INDT_NSN3,1,13)
    LSN4=SUBSTR(INDT_NSN4,1,13)
    LSN5=SUBSTR(INDT_NSN5,1,13)
    LSN6=SUBSTR(INDT_NSN6,1,13)
    LSN7=SUBSTR(INDT_NSN7,1,13)
    LSN8=SUBSTR(INDT_NSN8,1,13)
    IND='2'
    TQPL=QPA
    SELECT INDT
    APPEND BLANK
    REPLACE SRU WITH TSN
    REPLACE LRU WITH LSN1

```

```

REPLACE QPL WITH TEMP->QPL1
DO WHILE VAL(IND) < 9
  IF LEN(TRIM(LSN&IND)) > 0
    APPEND BLANK
    REPLACE SRU WITH TSN
    REPLACE LRU WITH LSN&IND
    REPLACE QPL WITH TEMP->QPL&IND
  ENDIF
  IND=TRIM(LTRIM(STR(VAL(IND)+1)))
ENDDO
SELE TEMP
LINE1=NSN+"      1 "+STR(QPA,3,0)+"
"+STR(DMD_RATE,7,5)+STR(DMD_RATE,7,5)+STR(BRCT,5,2)+"
"+STR(NRTS_RT,4,2)+" "+"0000"+"0000000000000000"
LINE2=NSN+" 0000 0000 0000000000      0000000000000000 "
?LINE1
?LINE2
SKIP
ENDDO

*
*INDT
*

SELE INDT
GO TOP
SET SAFE OFF
INDEX ON LRU TO LRU
SET SAFE ON
?"INDT"
IF RECCOUNT () >0
  LINE=LRU+"1L"
  ?LINE
  PLRU=LRU
  DO WHILE .NOT. EOF()
    IF .NOT. (LRU=PLRU)
      LINE=LRU+"1L"
      ?LINE
      PLRU=LRU
    ENDIF
    LINE1=SRU+" S"+STR(QPL,3)
    ?LINE1
    PSRU=TRIM(SRU)
    SELE TEMP
    SEEK TRIM(PLRU)
    IF .NOT. FOUND()
      MESSAGE=5
      CLEAR
      @ 5,33 SAY "*** WARNING ***"
      @ 7,0 SAY "SRU "+PSRU+"'s LRU "+PLRU+"HAS BEEN DELETED OR NOT
REVIEWED"
      @ 9,0 SAY "THIS WILL CAUSE FATAL ERRORS IN PROCESSING"
      STORE " " TO ANSW
      @ 15,10 SAY "DO YOU WANT TO CONTINUE ? (Y/N)";
      GET ANSW PICTURE "@!"
      READ
      IF ANSW = "N"

```



```

        USE
        SELE 1
        SET DELE OFF
        SET ALTE TO
        RETURN
    ENDIF
ELSE
    IF RR_RRR<>1
        SELE 2
        REPLACE ERROR2 WITH "Y"
        MESSAGE=5
        SELE 3
    ENDIF
    IF LRU_SRU="S"
        SELE 2
        REPLACE ERROR2 WITH "L"
        MESSAGE=5
        SELE 3
    ENDIF
    TMDR=DMD_RATE
    SEEK PSRU
    SMDR=DMD_RATE
    IF SMDR>TMDR
        SET ALTE OFF
        CLEAR
        @10,1 SAY "WARNING: THE DEMAND RATE FOR SRU "+PSRU+" IS
GREATER THAN ITS PARENT"
        @11,1 SAY "LRUs "+PLRU+" DEMAND RATE. THIS SHOULD NOT
HAPPEN. CHECK THESE NSNs DEMAND RATES."
        STORE " " TO ANSW
        @ 15,10 SAY "DO YOU WANT TO CONTINUE ? (Y/N)";
        GET ANSW PICTURE "@!";
        READ
        IF ANSW = "N"
            USE
            SELE 1
            SET DELE OFF
            SET ALTE TO
            RETURN
        ENDIF
        SET ALTE ON
    ENDIF
ENDIF
SELE INDT
SKIP
ENDDO
ENDIF

*
*   STK LEVEL
*
SELE TEMP
GO TOP
?"STK"
DO WHILE .NOT.EOF()
    LINE1=NSN+" "+BASE+" "+STR(STK,3)
    ?LINE1

```

```

SKIP
ENDDO
?"END"

```

```

SET ALTE OFF
*: EOF: DMV6OUT.PRG

```

```

*:*****
*
*:
*:      Program: MYREAD.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/24/95      17:32
*:
*:      Called by: BUTIL.PRG
*:
*:      Uses: BLNK.DBF
*:            : &TXT
*:
*:      Documented: 07/16/95 at 04:44                      SNAP! version
3.10a
*:*****
*
SELE 5
USE BLNK
SET SAFE OFF
ZAP
SET SAFE ON

SELE 1
SET SAFE OFF
ZAP
SET SAFE ON
SELE 5

APPE FROM &TXT SDF

GO TOP
HEADER=[LRU]
DO WHILE .NOT. EOF()

    IF SUBSTR(LINE,1,3)<>HEADER
        SKIP
        LOOP
    ELSE
        SKIP
        HEADER=[APPL]
        DO WHILE SUBSTR(LINE,1,4) <> HEADER
            RNSN=SUBSTR(LINE,1,16)
            RDEPOT=SUBSTR(LINE,18,4)
            RQPA=VAL(SUBSTR(LINE,26,3))
            RDMD=VAL(SUBSTR(LINE,34,7))
            RBRC=VAL(SUBSTR(LINE,48,5))

```

```

RNRT=VAL (SUBSTR (LINE, 54, 4))
SKIP
RCOS=VAL (SUBSTR (LINE, 58, 8))
RWUC=SUBSTR (LINE, 67, 7)
RCAN=VAL (SUBSTR (LINE, 75, 1))
SELE 1
APPE BLANK
REPLACE NSN WITH RNSN
REPLACE DEPOT WITH RDEPOT
REPLACE QPA WITH RQPA
REPLACE DMD_RATE WITH RDMD
REPLACE BRCT WITH RBRC
REPLACE NRTS_RT WITH RNRT
REPLACE COST WITH RCOS
REPLACE WUC WITH RWUC
REPLACE CANN_IND WITH RCAN
REPLACE LRU_SRU WITH "L"
SELE 5
SKIP
LOOP
ENDDO
SKIP
HEADER=[VTM]
DO WHILE SUBSTR (LINE, 1, 3) <> HEADER
  RNSN=SUBSTR (LINE, 1, 16)
  RAPP=VAL (SUBSTR (LINE, 22, 5))
  SELE 1
  SEEK RNSN
  REPLACE APP1 WITH RAPP
  SELE 5
  SKIP
  LOOP
ENDDO
SKIP
HEADER=[SRU]
DO WHILE SUBSTR (LINE, 1, 3) <> HEADER
  RNSN=SUBSTR (LINE, 1, 16)
  RRRR=VAL (SUBSTR (LINE, 18, 1))
  RWAD=VAL (SUBSTR (LINE, 20, 4))
  SELE 1
  SEEK RNSN
  REPLACE RR_RRR WITH RRRR
  REPLACE WADJ WITH RWAD
  SELE 5
  SKIP
  LOOP
ENDDO
SKIP
HEADER=[INDT]
DO WHILE SUBSTR (LINE, 1, 4) <> HEADER
  RNSN=SUBSTR (LINE, 1, 16)
  RDEPOT=SUBSTR (LINE, 18, 4)
  RQPA=VAL (SUBSTR (LINE, 26, 3))
  RDMD=VAL (SUBSTR (LINE, 34, 7))
  RBRC=VAL (SUBSTR (LINE, 48, 5))
  RNRT=VAL (SUBSTR (LINE, 54, 4))
  SKIP

```

```

RCOS=VAL (SUBSTR (LINE,58,8))
RWUC=SUBSTR (LINE,67,7)
SELE 1
APPE BLANK
REPLACE NSN WITH RNSN
REPLACE DEPOT WITH RDEPOT
REPLACE QPA WITH RQPA
REPLACE DMD_RATE WITH RDMD
REPLACE BRCT WITH RBRC
REPLACE NRTS_RT WITH RNRT
REPLACE COST WITH RCOS
REPLACE WUC WITH RWUC
REPLACE LRU_SRU WITH "S"
SELE 5
SKIP
LOOP
ENDDO
SKIP
HEADER=[STK]
DO WHILE SUBSTR (LINE,1,3) <> HEADER
  IF SUBSTR (LINE,18,1)='L'
    RNSN=SUBSTR (LINE,1,16)
    SKIP
    LOOP
  ENDIF
  SNSN=SUBSTR (LINE,1,16)
  RQPA=VAL (SUBSTR (LINE,19,3))
  SELE 1
  SEEK SNSN
  REPLACE INDT_NSN1 WITH RNSN
  REPLACE QPL1 WITH RQPA
  SELE 5
  SKIP
ENDDO
SKIP
HEADER=[INDC]
DO WHILE SUBSTR (LINE,1,4) <> HEADER
  RNSN=SUBSTR (LINE,1,16)
  RSTK=VAL (SUBSTR (LINE,22,5))
  SELE 1
  SEEK RNSN
  REPLACE STK WITH RSTK
  SELE 5
  SKIP
  LOOP
ENDDO
SKIP
HEADER=[END]
DO WHILE SUBSTR (LINE,1,3) <> HEADER
  RNSN=SUBSTR (LINE,1,13)
  RWUC=SUBSTR (LINE,16,5)
  RNOU=SUBSTR (LINE,23,10)
  RCOM=SUBSTR (LINE,33,3)
  RWAD=VAL (SUBSTR (LINE,36,4)) /100
  RNC=SUBSTR (LINE,40,1)
  SELE 1

```

```

        SEEK RNSN
        REPLACE WUC WITH RWUC
        REPLACE NOUN WITH RNOU
        REPLACE COMP WITH RCOM
        REPLACE WADJ WITH RWAD
        REPLACE NC WITH RNC
        SELE 5
        SKIP
        LOOP
    ENDDO
    SKIP

ENDIF

ENDDO

SELE 1
*: EOF: MYREAD.PRG

*: *****
*
*:
*:      Program: NSN.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/24/95      17:21
*:
*:      Called by: REPORT.PRG
*:
*:      Documented: 07/16/95 at 04:44                      SNAP! version
3.10a
*: *****
*
IF PRNT= 'Y'
    SET DEVICE TO PRINT
    LINE=60
ELSE
    CLEAR
    LINE=20
ENDIF
GO TOP
DO WHILE .NOT. EOF()
    @ 3, 10 SAY "NSN          WUC          &TAPP1  &TAPP2  &TAPP3  &TAPP4
&TAPP5  &TAPP6  &TAPP7  &TAPP8"
    IND=4
    DO WHILE (IND<LINE .AND. (.NOT. EOF()))
        @ IND, 5  SAY NSN
        @ IND, 23 SAY WUC
        IF LEN(TRIM(TAPP1))>0
            @ IND, 31 SAY APP1
        ENDIF
        IF LEN(TRIM(TAPP2))>0
            @ IND, 37 SAY APP2

```

```

ENDIF
IF LEN(TRIM(TAPP3))>0
  @ IND, 43 SAY APP3
ENDIF
IF LEN(TRIM(TAPP4))>0
  @ IND, 49 SAY APP4
ENDIF
IF LEN(TRIM(TAPP5))>0
  @ IND, 55 SAY APP5
ENDIF
IF LEN(TRIM(TAPP6))>0
  @ IND, 61 SAY APP6
ENDIF
IF LEN(TRIM(TAPP7))>0
  @ IND, 67 SAY APP7
ENDIF
IF LEN(TRIM(TAPP8))>0
  @ IND, 73 SAY APP8
ENDIF
IND=IND+1
SKIP
ENDDO
IF PRNT<>"Y"
  @ 21, 5 SAY "HIT ANY KEY TO CONTINUE"
  WAIT " "
ENDIF
CLEAR
ENDDO
IF PRNT= 'Y'
  SET DEVICE TO PRINT
  LINE=60
ELSE
  CLEAR
  LINE=20
ENDIF
GO TOP
IF LEN(TRIM(TAPP9))>0
  DO WHILE .NOT. EOF()
    @ 3, 10 SAY "NSN" WUC &TAPP9 &TAPP10 &TAPP11
    &TAPP12"
    IND=4
    DO WHILE (IND<LINE .AND. (.NOT. EOF()))
      @ IND, 5 SAY NSN
      @ IND, 23 SAY WUC
      IF LEN(TRIM(TAPP9))>0
        @ IND, 31 SAY APP9
      ENDIF
      IF LEN(TRIM(TAPP10))>0
        @ IND, 37 SAY APP10
      ENDIF
      IF LEN(TRIM(TAPP11))>0
        @ IND, 43 SAY APP11
      ENDIF
      IF LEN(TRIM(TAPP12))>0
        @ IND, 49 SAY APP12
      ENDIF
      IF LEN(TRIM(TAPP13))>0

```

```

        @ IND, 55 SAY APP13
    ENDIF
    IF LEN(TRIM(TAPP14))>0
        @ IND, 61 SAY APP14
    ENDIF
    IF LEN(TRIM(TAPP15))>0
        @ IND, 66 SAY APP15
    ENDIF
    IND=IND+1
    SKIP
ENDDO
IF PRNT<>"Y"
    @ 21, 5 SAY "HIT ANY KEY TO CONTINUE"
    WAIT " "
ENDIF
CLEAR
ENDDO
ENDIF
SET DEVICE TO SCREEN

```

```

*: EOF: NSN.PRG

```

```

*: *****
*
*:
*:      Program: READDMAS.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/24/95      17:21
*:
*:      Called by: BUTIL.PRG
*:
*:      Uses: DMASQTY.DBF
*:           : &NAME
*:
*:      Indexes: DMASQTY.NDX
*:
*:      Documented: 07/16/95 at 04:44      SNAP! version
3.10a
*: *****
*

```

```

SELE 3
USE DMASQTY
SET SAFE OFF
ZAP
SET SAFE ON
CNT=1
DO WHILE CNT>0
    NAME=[      ]
    ANS=[Y]
    CLEAR
    @12,5 SAY "GIVE THE DMAS QTY FILE NAME." GET NAME

```

```

@13,5 SAY "THIS MUST HAVE A .TXT EXTENSION."
@15,5 SAY "WERE NOPS EXCLUDED ?" GET ANS
READ
IF NAME=" "
    RETURN
ENDIF
NAME=(TRIM(LTRIM(NAME)))+".TXT"
IF .NOT. FILE(NAME)
    CLEAR
    @ 12, 5 SAY "The file &NAME does not exist."
    @ 23,0 SAY "PRESS ANY KEY TO CONTINUE..."
    WAIT " "
    LOOP
ELSE
    CNT=0
ENDIF
ENDDO
CLEAR
APPE FROM &NAME SDF
GO TOP
SET SAFE OFF
DELE FOR STKCLASS<[0000].OR.STKCLASS>[9999]
PACK
REPLACE ALL NSN WITH STKCLASS+NIIN

INDEX ON NSN TO DMASQTY
SET SAFE ON
SELE 1
IF ANS="Y"
    REPLACE ALL AUTHQTY WITH STK FOR COMP="NOP"
ENDIF
UPDATE ON NSN FROM DMASQTY REPLACE STK WITH DMASQTY->COMPQTY
IF ANS="Y"
    REPLACE ALL STK WITH AUTHQTY FOR COMP="NOP"
ENDIF
SELE 3
SET SAFE OFF
*ZAP
SET SAFE ON
USE
SELE 1

```

```

*: EOF: READDMAS.PRG

```

```

*:*****
*
*:
*:      Program: REPORT.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/24/95      17:21

```



```

*:
*:      Called by: BASEAID.PRG
*:
*:      Calls: SORT.PRG
*:            : NSN.PRG
*:
*:      Report Forms: FULLREP.FRM
*:                  : INDTREP.FRM
*:                  : INDTREP2.FRM
*:
*:      Documented: 07/16/95 at 04:44                      SNAP! version
3.10a
*:*****
*
```

DO WHILE .T.

```

* ---Display menu options, centered on the screen.
*   draw menu border and print heading
CLEAR
@ 2, 0 TO 13,79 DOUBLE
@ 3,34 SAY [R E P O R T S]
@ 4,1 TO 4,78 DOUBLE
* ---display detail lines
@ 7,23 SAY [1. LIST NSN & APPLICATION FRACTIONS]
@ 8,23 SAY [2. NSN INFORMATION]
@ 9,23 SAY [3. INDENTURE RELATIONSHIPS]
@ 11, 23 SAY '0. EXIT'
STORE 0 TO SELECTNUM
@ 13,33 SAY " select      "
@ 13,42 GET SELECTNUM PICTURE "9" RANGE 0,3
READ
```

```

DO CASE
CASE SELECTNUM = 0
    RETURN
```

```

CASE SELECTNUM = 1
    * DO LIST NSN & APLPLICATION FRACTIONS
    DO SORT
    DO NSN
    USE
    SELE &PDB
```

```

CASE SELECTNUM = 2
    DO SORT
    IF PRNT= 'Y'
        CLEAR
        @ 10,5 SAY "PLEASE SET PRINTER TO 17 PITCH"
        @ 11,5 SAY "PRESS ANY KEY TO CONTINUE"
        WAIT " "
        REPORT FORM FULLREP TO PRINT
    ELSE
        REPORT FORM FULLREP
        @ 24,5 SAY "PRESS ANY KEY TO CONTINUE"
        WAIT " "
    ENDIF
```

```

USE
SELE &PDB

CASE SELECTNUM = 3
  * DO INDENTURE RELATIONSHIPS
DO SORT
IF PRNT= 'Y'
  CLEAR
  @ 10,5 SAY "PLEASE SET PRINTER TO 17 PITCH"
  @ 11,5 SAY "PRESS ANY KEY TO CONTINUE"
  WAIT " "
  REPORT FORM INDREP FOR LRU_SRU="S" TO PRINT
  GO TOP
DO WHILE .NOT. EOF()
  IF LEN(TRIM(INDT_NSN5))>0
    REPORT FORM INDREP2 FOR LRU_SRU="S" TO PRINT
    EXIT
  ELSE
    SKIP
    LOOP
  ENDIF
ENDDO
ELSE
  REPORT FORM INDREP FOR LRU_SRU="S"
  @ 24,5 SAY "PRESS ANY KEY TO CONTINUE"
  WAIT " "
  GO TOP
DO WHILE .NOT. EOF()
  IF LEN(TRIM(INDT_NSN5))>0
    REPORT FORM INDREP2 FOR LRU_SRU="S"
    @ 24,5 SAY "PRESS ANY KEY TO CONTINUE"
    WAIT " "
    EXIT
  ELSE
    SKIP
    LOOP
  ENDIF
ENDDO
ENDIF
USE
SELE &PDB

ENDCASE

ENDDO T
RETURN
* EOF: REPORT.PRG

*: EOF: REPORT.PRG

*: *****
*
*:
*:
*:      Program: SIMOUT.PRG

```

```

*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 07/16/95      4:36
*:
*:      Called by: BBUILD.PRG
*:              : SORTCOMB.PRG
*:
*:      Uses: INDT.DBF
*:
*:      Indexes: LRU.NDX
*:
*:      Documented: 07/16/95 at 04:44      SNAP! version
3.10a
*:*****
*

```

```

GO TOP
SELECT 2
USE INDT
SET SAFETY OFF
ZAP
SET SAFETY ON
SELECT TEMP

DO WHILE .NOT. EOF()
  IF .NOT. (LRU_SRU="S")
    SKIP
    LOOP
  ENDIF
  TSN=NSN
  LSN1=SUBSTR(INDT_NSN1,1,13)
  LSN2=SUBSTR(INDT_NSN2,1,13)
  LSN3=SUBSTR(INDT_NSN3,1,13)
  LSN4=SUBSTR(INDT_NSN4,1,13)
  LSN5=SUBSTR(INDT_NSN5,1,13)
  LSN6=SUBSTR(INDT_NSN6,1,13)
  LSN7=SUBSTR(INDT_NSN7,1,13)
  LSN8=SUBSTR(INDT_NSN8,1,13)
  IND='2'
  TQPL=QPA
  SELECT INDT
  APPEND BLANK
  REPLACE SRU WITH TSN
  REPLACE LRU WITH LSN1
  REPLACE QPL WITH TEMP->QPL1
  DO WHILE VAL(IND) < 9
    IF LEN(TRIM(LSN&IND)) > 0
      APPEND BLANK
      REPLACE SRU WITH TSN
      REPLACE LRU WITH LSN&IND
      REPLACE QPL WITH TEMP->QPL&IND
    ENDIF
    IND=TRIM(LTRIM(STR(VAL(IND)+1)))
  ENDDO
SELE TEMP

```

```

        SKIP
    ENDDO

    SELE INDT
    GO TOP
    SET SAFE OFF
    INDEX ON LRU TO LRU
    SET SAFE ON
    IF RECCOUNT ( ) >0
        PLRU=LRU
        SRUCNT=0
        DO WHILE .NOT. EOF()
            IF .NOT. (LRU=PLRU)
                SELE TEMP
                SEEK TRIM(PLRU)
                IF FOUND()
                    REPLACE SRUCOUNT WITH SRUCNT
                ENDIF
                SRUCNT=0
                SELE INDT
                PLRU=LRU
            ENDIF
            SRUCNT=SRUCNT+1
            PSRU=TRIM(SRU)
            SELE TEMP
            SEEK TRIM(PLRU)
            IF .NOT. FOUND()
                MESSAGE=5
                CLEAR
                @ 5,33 SAY "*** WARNING ***"
                @ 7,0 SAY "SRU "+PSRU+"'s LRU "+PLRU+"HAS BEEN DELETED OR NOT
REVIEWED"
                @ 9,0 SAY "THIS WILL CAUSE FATAL ERRORS IN PROCESSING"
                STORE " " TO ANSW
                @ 15,10 SAY "DO YOU WANT TO CONTINUE ? (Y/N)";
                GET ANSW PICTURE "@!"
                READ
                IF ANSW = "N"
                    USE
                    SELE 1
                    SET DELE OFF
                    SET ALTE TO
                    RETURN
                ENDIF
            ELSE
                IF RR_RRR<>1
                    SELE 2
                    REPLACE ERROR2 WITH "Y"
                    MESSAGE=5
                    SELE 3
                ENDIF
                IF LRU_SRU="S"
                    SELE 2
                    REPLACE ERROR2 WITH "L"
                    MESSAGE=5
                    SELE 3
                ENDIF
            ENDIF
        ENDWHILE
    ENDIF

```

```

ENDIF
TMDR=DMD_RATE
SEEK PSRU
SMDR=DMD_RATE
IF SMDR>TMDR
    SET ALTE OFF
    CLEAR
    @10,1 SAY "WARNING: THE DEMAND RATE FOR SRU "+PSRU+" IS
GREATER THAN ITS PARENT"
    @11,1 SAY "LRUs "+PLRU+" DEMAND RATE. THIS SHOULD NOT
HAPPEN. CHECK THESE NSNs DEMAND RATES."
    STORE " " TO ANSW
    @ 15,10 SAY "DO YOU WANT TO CONTINUE ? (Y/N)";
    GET ANSW PICTURE "@!";
    READ
    IF ANSW = "N"
        USE
        SELE 1
        SET DELE OFF
        SET ALTE TO
        RETURN
    ENDIF
    SET ALTE ON
ENDIF
ENDIF
SELE INDT
SKIP
ENDDO
SELE TEMP
SEEK TRIM(PLRU)
IF FOUND()
    REPLACE SRUCOUNT WITH SRUCNT
ENDIF
ENDIF

SELE TEMP
GO TOP
LRUCNT=0
DMTOT=0
DO WHILE .NOT.EOF()
    IF .NOT.(LRU_SRU="L")
        SKIP
        LOOP
    ENDIF
    LRUCNT=LRUCNT+1
    DMTOT=DMTOT+DMD_RATE*QPA*APP&APPNO

    SKIP
ENDDO

SET ALTE TO NLIP.INP
SET ALTE ON
?DMTOT
?"40"
?LRUCNT
?"1"
I=1

```

```

DO WHILE I<41
    ?I
    I=I+1
ENDDO
?I
SET ALTE OFF
!NLIP
!TYPE NLIP.OUT
WAIT "THESE ARE THE OPT # OF BATCHES & BATCH SIZE. PRESS ANY KEY"
CLEAR
BATCH1=[    ]
BATCH2=[    ]
ACNO=[    ]
FLSOR=[    ]
SORTDAY=[    ]
@ 5,5 SAY "WHAT BATCH SIZE DO YOU WANT" GET BATCH1
@ 8,5 SAY "HOW MANY BATCHES" GET BATCH2
@ 11,5 SAY "HOW MANY AIRCRAFT" GET ACNO
@ 14,5 SAY "HOW MANY FLYING HOURS PER SORTIE" GET FLSOR
@ 17,5 SAY "HOW MANY SORTIES PER DAY" GET SORTDAY
READ
MINAC=INT (VAL (ACNO) -1)
GO TOP

SET ALTE TO &FTXT

SET ALTE ON

?DMTOT
?BATCH2
?BATCH1
?ACNO
?MINAC
?LRUCNT
?SORTDAY
?FLSOR
?"0"

DO WHILE .NOT.EOF()
    IF .NOT.(LRU_SRU="L")
        SKIP
        LOOP
    ENDIF
    DMD_RATE=DMD_RATE*APP1
    LINE1=SUBSTR (NSN,1,13)+" "+STR(DMD_RATE,7,5)+" "+STR(QPA,3)+"
"+STR(MQPA,3)+" "+STR(STK,3)+" "+STR(SRUCOUNT,2)+"
"+STR(NRTS_RT,4,2)+STR(BRCT,5,2)
    ? LINE1
    DMD_RATE=DMD_RATE/APP1
    SKIP
ENDDO

SET ALTE TO SRU.INP
SET ALTE ON
GO TOP
DO WHILE .NOT. EOF()

```

```

      IF .NOT. (LRU_SRU="S")
        SKIP
      LOOP
    ENDIF
    LINE1=SUBSTR(NSN,12,2)+" "+STR(DMD_RATE,7,5)+" "+STR(QPA,3)+"
"+STR(STK,3)
    ?LINE1
    SKIP
  ENDDO

```

```

SET ALTE OFF
*: EOF: SIMOUT.PRG

```

```

*: *****
*
*:
*:      Program: SORT.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/24/95      17:21
*:
*:      Called by: REPORT.PRG
*:
*:      Uses: TEMP.DBF
*:
*:      Indexes: NSN.NDX
*:                : WUC.NDX
*:                : TEMP.NDX
*:                : RR.NDX
*:                : SLRU.NDX
*:
*:      Documented: 07/16/95 at 04:44                      SNAP! version
3.10a
*: *****
*

```

```

* ---Display menu options, centered on the screen.
*      draw menu border and print heading
CLEAR
@ 2, 0 TO 15,79 DOUBLE
@ 3,24 SAY [S O R T S   F O R   R E P O R T S]
@ 4,1 TO 4,78 DOUBLE
* ---display detail lines
@ 7,26 SAY [1. SORT BY NSN]
@ 8,26 SAY [2. SORT BY WUC]
@ 9,26 SAY [3. SORT BY REVIEWED FLAG]
@ 10,26 SAY [4. SHOW ONLY NON-DELETED NSN]
@ 11,26 SAY [5. SORT BY RR & RRR]
@ 12,26 SAY [6. SORT BY LRU AND SRU]
STORE 1 TO SELECTNUM
@ 15,33 SAY " select      "
@ 15,42 GET SELECTNUM PICTURE "9" RANGE 1,6

```

READ

DO CASE
CASE SELECTNUM = 0
RETURN

CASE SELECTNUM = 1
* SORT BY NSN
SET SAFETY OFF
COPY TO TEMP
SELE 3
USE TEMP
INDEX ON NSN TO NSN
SET SAFE ON
RETURN

CASE SELECTNUM = 2
* SORT BY WUC
SET SAFETY OFF
COPY TO TEMP
SELE 3
USE TEMP
INDEX ON WUC TO WUC
SET SAFE ON

CASE SELECTNUM = 3
* SHOW ONLY REVIEWED NSN
SET SAFE OFF
CLEAR
TANS=[B]
@ 10, 5 SAY "DO YOU WANT ONLY (R)EVIEWED, (N)OT-REVIEWED, OR (B)OTH
?" GET TANS
READ
DO CASE
CASE TANS = [R]
COPY TO TEMP FOR REVIEWED=[Y]
CASE TANS = [N]
COPY TO TEMP FOR REVIEWED<>[Y]
OTHERWISE
COPY TO TEMP
ENDCASE
SELE 3
USE TEMP
PACK
INDEX ON REVIEWED+NSN TO TEMP
SET SAFE ON

CASE SELECTNUM = 4
* SHOW ONLY NON-DELETED NSN
SET SAFETY OFF
COPY TO TEMP
SELE 3
USE TEMP
PACK
INDEX ON NSN TO TEMP
SET SAFE ON


```

CASE SELECTNUM = 5
  * DO SORT BY RR & RRR
  SET SAFETY OFF
  COPY TO TEMP
  SELE 3
  USE TEMP
  INDEX ON RR_RRR TO RR
  SET SAFE ON

```

```

CASE SELECTNUM = 6
  * DO SORT BY LRU AND SRU
  SET SAFETY OFF
  COPY TO TEMP
  SELE 3
  USE TEMP
  INDEX ON LRU_SRU TO SLRU
  SET SAFE ON

```

```

ENDCASE

```

```

RETURN

```

```

* EOF: SORT.PRG
*: EOF: SORT.PRG

```

```

*:*****
*
*:
*:      Program: SORTCOMB.PRG
*:
*:      System: disc-data-handler
*:      Author: t.lewis
*:      Copyright (c) 1994, t.lewis
*:      Last modified: 06/28/95      20:22
*:
*:      Called by: BBUILD.PRG
*:
*:      Calls: SIMOUT.PRG
*:
*:      Uses: SORTCOMB.DBF
*:            : OUTPUT.TXT
*:
*:      Indexes: NSNINDE.NDX
*:                : TEMP.NDX
*:                : QPROB.NDX
*:
*:      Documented: 07/16/95 at 04:44
*:
*:
*:      SNAP! version
3.10a
*:*****
*
SET TALK OFF
USE SORTCOMB
SET SAFE OFF
ZAP
APPE FROM OUTPUT.TXT SDF
REPLACE ALL NIIN WITH SUBSTR(NSN,5,9)

```

```

INDE ON NIIN TO NSNINDE
REPLACE ALL QPROB WITH 1-QPROB
*RETURN
SELE TEMP
REPLACE ALL NIIN WITH SUBSTR(NSN,5,9)
INDE ON NIIN TO TEMP
SET SAFE ON
SET RELA TO NIIN INTO SORTCOMB
REPLACE ALL QPROB WITH SORTCOMB->QPROB FOR NIIN=SORTCOMB->NIIN
*RETURN

```

```

*C      This section sorts out the low demand/probability of failure
*C      items from the computations.

```

```

*C Will need way to sort the data by U(i,ac) values. (Smallest to
largest)

```

```

SET SAFE OFF
INDEX ON QPROB TO QPROB
SET SAFE ON
GO TOP
TOLERANC=[          ]
CLEAR
@10,5 SAY "Give the acceptable Tolerance" GET TOLERANC
READ
IF TOLERANC=[      ]
    RETURN
ENDIF
TOLERANC=VAL(TOLERANC)
CURTOL=1
DELETEVAR=RECCOUNT()
ACCURACY=1-TOLERANC
I=1
DO WHILE I<RECCOUNT()
    CURTOL=CURTOL*(1-QPROB)
    IF ACCURACY<CURTOL
        *C---- these two lines unnecessary. should delete the NSN!!
        QPROB=1
        *****      DELETE
    ELSE
        *c---   delete the first i-1 items or do it above but remember
counter!
        DELETEVAR=I-1
        CURTOL=CURTOL/(1-QPROB)
        I=RECCOUNT()
    ENDIF
    I=I+1
    SKIP
ENDDO

PACK
CURTOL=1-CURTOL
*C      Prints the number of items deleted and the accuracy loss
?"the number deleted and accuracy loss"
? DELETEVAR,CURTOL

```

```

WAIT" "
SET SAFE OFF
INDE ON SUBSTR(NSN,5,9) TO TEMP
SET SAFE ON
SET ALTE TO &FTXT
DO SIMOUT
SELE 3
USE
SELE 2
SET SAFE OFF
ZAP
SET SAFE ON
SET PRINT OFF
SET ALTE TO
SET DELE OFF

```

*: EOF: SORTCOMB.PRG

This is a listing of the data structure for the primary database used in this system.

Structure for database: C:\wrskblnk.dbf

Number of data records: 0

Date of last update : 07/16/95

Field	Field Name	Type	Width	Dec
1	NSN	Character	16	
2	WUC	Character	5	
3	QPA	Numeric	2	
4	DMD_RATE	Numeric	7	5
5	NOUN	Character	25	
6	LRU_SRU	Character	1	
7	RR_RRR	Numeric	1	
8	COST	Numeric	8	
9	COMP	Character	3	
10	AUTHQTY	Numeric	3	
11	INDT_NSN1	Character	16	
12	QPL1	Numeric	2	
13	INDT_NSN2	Character	16	
14	QPL2	Numeric	2	
15	INDT_NSN3	Character	16	
16	QPL3	Numeric	2	
17	INDT_NSN4	Character	16	
18	QPL4	Numeric	2	
19	INDT_NSN5	Character	16	
20	QPL5	Numeric	2	
21	INDT_NSN6	Character	16	
22	QPL6	Numeric	2	
23	INDT_NSN7	Character	16	
24	QPL7	Numeric	2	
25	INDT_NSN8	Character	16	
26	QPL8	Numeric	2	
27	REVIEWED	Character	1	
28	DEL	Character	1	
29	BRCT	Numeric	6	2

30	NRTS_RT	Numeric	5	2
31	CONDM_RT	Numeric	5	2
32	CANN_IND	Numeric	2	
33	NRTS_CONDM	Numeric	1	
34	WADJ	Numeric	6	
35	MTBD	Numeric	6	
36	NC	Character	1	
37	BASE	Character	4	
38	DEPOT	Character	2	
39	IM	Character	3	
40	REMARK	Character	80	
41	REMARKS	Character	80	
42	APP1	Numeric	4	2
43	NOP1	Numeric	3	
44	POS1	Numeric	6	
45	APP2	Numeric	4	2
46	NOP2	Numeric	3	
47	POS2	Numeric	6	
48	APP3	Numeric	4	2
49	NOP3	Numeric	3	
50	POS3	Numeric	6	
51	APP4	Numeric	4	2
52	NOP4	Numeric	3	
53	POS4	Numeric	6	
54	APP5	Numeric	4	2
55	NOP5	Numeric	3	
56	POS5	Numeric	6	
57	APP6	Numeric	4	2
58	NOP6	Numeric	3	
59	POS6	Numeric	6	
60	APP7	Numeric	4	2
61	NOP7	Numeric	3	
62	POS7	Numeric	6	
63	APP8	Numeric	4	2
64	NOP8	Numeric	3	
65	POS8	Numeric	6	
66	APP9	Numeric	4	2
67	NOP9	Numeric	3	
68	POS9	Numeric	6	
69	APP10	Numeric	4	2
70	NOP10	Numeric	3	
71	POS10	Numeric	6	
72	APP11	Numeric	4	2
73	NOP11	Numeric	3	
74	POS11	Numeric	6	
75	APP12	Numeric	4	2
76	NOP12	Numeric	3	
77	POS12	Numeric	6	
78	APP13	Numeric	4	2
79	NOP13	Numeric	3	
80	POS13	Numeric	6	
81	APP14	Numeric	4	2
82	NOP14	Numeric	3	
83	POS14	Numeric	6	
84	APP15	Numeric	4	2
85	NOP15	Numeric	3	
86	POS15	Numeric	6	

87	APP16	Numeric	4	2
88	NOP16	Numeric	3	
89	APP17	Numeric	4	2
90	NOP17	Numeric	3	
91	POS17	Numeric	6	
92	APP18	Numeric	4	2
93	NOP18	Numeric	3	
94	APP19	Numeric	4	2
95	NOP19	Numeric	3	
96	APP20	Numeric	4	2
97	NOP20	Numeric	3	
98	APPL	Numeric	4	2
99	STK	Numeric	3	
100	BASE1DMD	Numeric	5	
101	BASE2DMD	Numeric	5	
102	BASE3DMD	Numeric	5	
103	BASE4DMD	Numeric	5	
104	BASE5DMD	Numeric	5	
105	BASE6DMD	Numeric	5	
106	BASE7DMD	Numeric	5	
107	BASE8DMD	Numeric	5	
108	BASE9DMD	Numeric	5	
109	BASE10DMD	Numeric	5	
110	HOURS	Character	1	
111	CMDMTBD	Numeric	9	2
112	TOTDMS	Numeric	6	
113	TOTFLYHRS	Numeric	9	
114	DATA_2YRS	Character	15	
115	DATA_1YRS	Character	15	
116	DATA_CURR	Character	15	
117	DATA_3YRS	Character	15	
118	DO41MTBD	Numeric	6	
119	DO41QPA	Numeric	3	
120	DO41BRCT	Numeric	6	2
121	DO41NRTS	Numeric	5	2
122	MQPA	Numeric	2	
123	POS	Numeric	6	
124	UPDATE	Character	1	
125	MDR	Numeric	7	5
126	NIIN	Character	9	
127	QPROB	Numeric	17	15
128	SRUCOUNT	Numeric	2	
** Total **			861	